(2)

Center for Human-Machine Systems Research

An ICAI Architecture for Troubleshooting in Complex Dynamic Systems

Janet L. Fath
Christine M. Mitchell
T. Govindaraj

Report 88-4

December 1988

School of Industrial and Systems Engineering
Georgia Institute of Technology
A Unit of the University System of Georgia
Atlanta, Georgia 30332-0205

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) 88-4	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Center for Human-Machine Systems Research, Ga. Inst. of Tech.	6b. OFFICE SYMBOL (If applicable)
6c. ADDRESS (City, State, and ZIP Code) School of Industrial and Systems Engineering 765 Ferst Drive Atlanta, GA 30332-0205	7a. NAME OF MONITORING ORGANIZATION Cognitive Science Program Office of Naval Research (Code 1142CS)
	7b. ADDRESS (City, State, and ZIP Code) 800 North Quincy Street Arlington, VA 22217-5000
8a. NAME OF FUNDING / SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (If applicable)
	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-87-K-0482
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO
62233N	RM33M20		

11. TITLE (Include Security Classification)
An ICAI Architecture for Troubleshooting in Complex Dynamic Systems

12. PERSONAL AUTHOR(S) Janet L. Fath, Christine M. Mitchell, T. Govindaraj

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM _____ TO _____	1988 December	

16. SUPPLEMENTARY NOTATION
Supported by the Office of the Chief of Naval Research, Manpower, Personnel, and Training R&D Program. Partial funding from NASA Goddard Space Flight Center

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP
23	02	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
fault diagnosis; problem solving; training; intelligent tutoring systems; intelligent computer assisted instruction

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

AHAB is an architecture for simulator-based ICAI programs to teach troubleshooting in complex, dynamic environments. The architecture posits three elements of a computerized instructor: the task model, the student model, and the instructional module. The task model is a prescriptive model of expert performance that uses symptomatic and topographic search strategies to provide students with directed problem-solving aids. The student model is a descriptive model of student performance in the context of the task model. This student model compares the student and task models, critiques student performance, and provides interactive performance feedback. Finally, the instructional module coordinates information presented by the instructional media, the task model, and the student model so that each student receives individualized instruction. Concept and metaconcept knowledge that supports these elements is contained in frames and production rules, respectively.

The results of an experimental evaluation support the hypothesis that training with an adaptive online system built using the AHAB architecture produces better performance than training using simulator practice alone, at least with unfamiliar problems. Furthermore, it is not sufficient to develop an expert strategy and present it to students using offline materials. The training is most effective if it adapts to individual student needs.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Susan E. Chipman	22b. TELEPHONE (Include Area Code) 202-696-4318
	22c. OFFICE SYMBOL ONR 1142 CS

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

An ICAI Architecture for Troubleshooting in Complex Dynamic Systems

Janet L. Fath*
Christine M. Mitchell
T. Govindaraj

Center for Human-Machine Systems Research
School of Industrial & Systems Engineering
Georgia Institute of Technology
Atlanta, Ga 30332
(404) 894-2300

Submitted for publication to *IEEE Transactions on Systems, Man, and Cybernetics*

* Dr. Fath is currently with the IBM Corp, Application Systems Division, 1500 River Edge Pkwy. R19A, Atlanta, GA

This research was supported in part by the Office of Naval Research, contract N00014-82-K-0487, Work Unit NR 154-491 and by NASA Goddard contract NAS5-28575.

Abstract

AHAB is an architecture for simulator-based ICAI programs to teach troubleshooting in complex, dynamic environments. The architecture posits three elements of a computerized instructor: the task model, the student model, and the instructional module. The task model is a prescriptive model of expert performance that uses symptomatic and topographic search strategies to provide students with directed problem-solving aids. The student model is a descriptive model of student performance in the context of the task model. This student model compares the student and task models, critiques student performance, and provides interactive performance feedback. Finally, the instructional module coordinates information presented by the instructional media, the task model, and the student model so that each student receives individualized instruction. Concept and metaconcept knowledge that supports these elements is contained in frames and production rules, respectively.

The results of an experimental evaluation support the hypothesis that training with an adaptive online system built using the AHAB architecture produces better performance than training using simulator practice alone, at least with unfamiliar problems. Furthermore, it is not sufficient to develop an expert strategy and present it to students using offline materials. The training is most effective if it adapts to individual student needs.

Introduction

The need for better methods of training supervisory controllers has motivated exploration of Intelligent Computer Aided Instruction (ICAI) programs. This need arises from increased complexity of supervisory control systems and from current training conditions that may not be adequate for the tasks of interest (Kratt and Govindaraj, 1984). ICAI programs are a viable training alternative since they are capable of managing the complexity associated with large-scale, dynamic systems and the tasks that are performed within them. Moreover, ICAI programs provide individualized instruction similar to that of a human instructor in a one-on-one interaction with a student.

Review of the ICAI literature reveals that relatively few ICAI programs have been built (Wenger, 1987). A reason for the small number of existing ICAI programs, especially for applications in large-scale, dynamic systems, is that ICAI program developers must have a thorough understanding of a range of diverse and often complex subjects, including the large-scale, dynamic system, the task or function of the human operator in the system, artificial intelligence-based knowledge representation techniques, expert systems, online human-performance modeling techniques, instructional strategies, and human-computer interface design (Clancey, 1987). In order to facilitate development of additional ICAI programs, therefore, an architecture is necessary for specifying the knowledge needed and the way to organize it. This paper proposes such an ICAI architecture for troubleshooting in complex, dynamic environments. The architecture does not eliminate the knowledge requirements for program developers, but it does organize the required knowledge for the troubleshooting task and represents goals the resulting ICAI program should achieve.

A characteristic of the architecture is that the ICAI program has access to a simulator of the large-scale, dynamic system on which the task is performed. A second characteristic is that the ICAI program contains three elements: task model, student model, and instructional module (Wenger, 1987). In the remaining portion of this section, advantages of simulator-based ICAI programs and the three elements of an ICAI system are discussed.

Advantages of Simulator-Based ICAI Programs

Simulators have long been used for instruction. They allow students to learn while performing a version of an actual task. Simulators also allow students to explore new system configurations, discover new concepts, and attempt new procedures without damaging costly equipment or creating safety hazards.

Both static and dynamic simulators are used in instructional programs. Static simulators are typically used to teach students concepts that do not change over time. For instance, Burton and Brown (1982) use a simulator of a board game called How the West was Won to teach mathematical concepts. Hunt and Rouse (1981) use a simulation of a steady state car engine to teach students diagnostic skills.

Dynamic simulators are used to teach students about time varying systems. Such simulators can provide students with the opportunity to observe a process in a way real systems do not allow. For example, STEAMER (Williams et al., 1981; Hollan et al., 1984; Stevens et al., 1983) shows through animation how various fluids travel through a steam propulsion plant. Observation of a real steam plant makes such a view impossible due to the large scale and complexity of the plant and the inability of humans to see inside of plant components such as pipes and motors.

Students "learn by doing" with simulator-based ICAI programs because they can practice their tasks on the simulator while receiving individualized instruction, or instruction that adapts to their current instructional needs. Such adaptive computer-assisted instruction includes explanations of the functional effects of the simulator, demonstrations of recommended problem-solving approaches, evaluation of students' strategies, and adaptive selection of problems (Towne, 1986). Instruction can be presented in the context of simulated, yet realistic situations. Additional benefits of simulator-based instructional programs include the following (Emmett, 1984):

1. A seemingly endless variety of problems, exercises, etc. can be generated.
2. The pace of the simulated process can be speeded up or slowed down to illustrate various concepts to the student.
3. Students can create and test hypotheses and models from the simulator data.

Elements of the Architecture

Figure 1 illustrates an instructional system that uses the proposed architecture. Figure 1 shows that, in addition to a simulator, an ICAI program developed with the proposed architecture contains three elements: 1) a task model, 2) a student model, and 3) an instructional module. The three elements and the related issue of knowledge representation are described below.

The task model is a prescriptive model that specifies dynamically the relationship between system states and actions. The purpose of the task model is to provide directed problem-solving aids so that students can see the way the task is supposed to be performed. This model prescribes actions based on the current system state and organizes knowledge hierarchically as concepts and metaconcepts. It represents two problem-solving strategies (Rasmussen, 1986): 1) symptomatic search, which maps symptoms to possible causes, and 2) topographic search, which maps patterns of normal operating conditions to patterns of abnormal operating conditions.

The student model is a descriptive model of the student that classifies student actions as either errors or correct actions according to the prescribed strategy of the task model. Using the student model, the ICAI program identifies correct and erroneous student actions and provides interactive performance feedback to students based on a comparison of the student's actions with those resulting from possible misconceptions or from correct knowledge as contained in the task model. The student model is primarily an overlay model since it contains a subset of the knowledge in the task model (Clancey, 1987). As in the task model, knowledge is organized as concepts and metaconcepts. The student model is also a limited bug model (Wenger, 1987; Clancey, 1987) since it attempts to explain a few selected bugs in a student's problem-solving strategy.

The purpose of the instructional module is to coordinate information from the task model, the student model, and instructional media (e.g., the simulator, quiz programs, and other instructional information) to make instructional management decisions. These decisions concern the instructional medium, curriculum, pace of instruction, amount of feedback, and degree of control students may exercise over the instruction. Decisions are based on instructional practices found in the literature. Production rules are used to make the decisions.

The issue of knowledge representation affects each of the three elements of the instructional system. Knowledge in the task and student models is represented from multiple viewpoints, at multiple levels, and at the correct grain size (Wenger, 1987). By representing knowledge from multiple viewpoints, directed problem-solving aids use multiple strategies to prescribe actions. Furthermore, students' actions are explained in terms of multiple strategies, instead of being forced into only one. Representing knowledge at multiple levels separates concepts from procedures and enables the instructional module to tailor instruction to specific student errors and abilities. Finally, representing knowledge at the correct grain size ensures that students are not expected to learn concepts that are too complex or too trivial and also affects the order in which topics are presented.

The architecture was applied to an instructional system for teaching Navy ROTC students to troubleshoot failures in a marine powerplant simulator. The simulator is called PEQUOD. AHAB is the name of the ICAI program that teaches students to troubleshoot failures in PEQUOD. In the next section, PEQUOD is described. Using AHAB as an example, the architecture is then described in detail. An experiment to test the quality of instruction provided by AHAB is then discussed.

PEQUOD

PEQUOD is a version of Q-STEAM1, a marine powerplant simulator (Govindaraj, 1987). This simulator models the engine that provides power and support operations on a large ocean-going ship. PEQUOD uses a qualitative approximation methodology that involves a combination of control-theoretic and qualitative reasoning models. Using this methodology, a marine powerplant is represented hierarchically in terms of components and subsystems. Each of the approximately 75 simulated components is described by one or more primitives (e.g., conduit, source, sink, heat exchanger) that is tuned to simulate the functional behavior of the associated real component. The five subsystems are defined by functionally related loops such as Water or Fuel. Although system states (i.e., pressures, temperatures, levels, and flows) are calculated quantitatively, gauge readings are presented in terms of qualitative values (e.g., high and low).

PEQUOD runs on a Xerox 1109 LISP machine. Students interact with the simulator via a graphical interface that depicts subsystem schematic diagrams. Students use a mouse to read gauges and to make diagnoses.

PEQUOD is dynamic. Although only one component fails in each problem, the effects of that failure propagate through the system over a period of time, as they do in real systems. To represent more clearly to students the step-wise propagation of failures over time, the system state is updated at 30 second intervals. Students have access to a clock so that they know the elapsed time (i.e., the number of intervals) since the onset of the failure.

As in a real marine powerplant, various fluids flow through PEQUOD's components. Components have varying effects on the fluids that pass through them. Some components, for example, change the state of the fluid that enters them. For instance, the Steam Drum changes liquid Water to gaseous Steam. Other components, i.e., valves and pumps, affect the quantity of fluid that travels downstream from them. Still other components merely serve as conduits through which a fluid may pass. For instance, the Stack allows Gas, the by-product of combustion, to be vented to the atmosphere.

Components in PEQUOD can be grouped into three main processes: 1) steam generation, 2) steam use, and 3) steam condensation. The processes that comprise in PEQUOD are summarized in Figure 2. Each of

these processes can be further broken down into subprocesses. For instance, the steam generation process is made up of the combustion and boiling and heating subprocesses. At the lowest level are the processes defined by the transport of a single fluid through a series of components. Examples of the lowest level processes are Fuel transport and Water transport.

Components in PEQUOD can also be grouped into five subsystems: 1) Fuel Oil, 2) Boiler, 3) Steam, 4) Gland Seal, and 5) Condensate. One or more of the three main processes (i.e., steam generation, steam use, and steam condensation) may occur in each subsystem. Each subsystem is represented on a schematic diagram that is available for operator use. An example schematic diagram is shown in Figure 3.

Some parts are represented on the schematic diagrams that are not system components. For instance, Atmosphere is included in the Boiler Subsystem in Figure 3. Atmosphere is included is to show where the FDB (Forced Draft Blower) gets Boiler Air and what happens to Gas from the Stack. The other type of non-component shows the relationship between a part in a given subsystem and an adjacent subsystem. A square box with a subsystem name in it indicates where another subsystem connects to the present one. For example, a part from the Condensate Subsystem connects to the Steam Drum in the Boiler Subsystem (Figure 3).

The schematics also show gauges that can be read. Gauge symbols are small circles with a letter F, L, P, S, or T inside. The letters stand for Flow, Level, Pressure, Smoke,[1] and Temperature, respectively. The Steam Drum (Figure 3), for example, has a level and a pressure gauge for Saturated Steam going to the Boiler Tubes, a pressure gauge for Gas going to the Economizer, and a temperature gauge for Water from the Economizer.

Students interact with the simulator to troubleshoot a failure. At the beginning of a problem, one component fails. Students are provided with initial symptom information related to the failure condition. Students are asked to locate the failed component as quickly as possible and by making as few tests (i.e., gauge readings and test diagnoses) as possible.

---

[1] Smoke gauges allow the troubleshooter to look in the boiler periscope to see the color of the Gas passing through the Stack.

The Architecture

AHAB, an ICAI program, was built to illustrate and evaluate the proposed architecture (Figure 1). AHAB teaches students to troubleshoot failures in PEQUOD (Fath, 1987). AHAB includes a task model, student model and instructional module to provide the student with directed problem-solving aids, interactive performance feedback, and adaptive instructional management. In addition, AHAB illustrates the architecture's approach to knowledge representation. Development of AHAB demonstrates the feasibility of using the architecture in an ICAI program for a real training situation. Details of the architecture are presented below.

Task Model

The function of the task model is to prescribe to the student an action or set of actions to perform in order to troubleshoot failures in the simulator. An action is defined as any instruction or request for information that a student makes in order to perform the task. The troubleshooting task requires students to locate failed components as quickly as possible. Prescribed actions differ depending on the current state of the system.

The task model has three characteristics: 1) it prescribes troubleshooting actions based on the current system state, 2) it contains strategies that students can learn, and 3) it represents knowledge as concepts (facts) and metaconcepts (processes). These characteristics are discussed below.

The first characteristic of the task model is that it prescribes troubleshooting actions based on the current system state. The task model is thus capable of accounting for a dynamically changing system. Furthermore, the actions that it prescribes are presented in a way that is readily understood by the student and directly applicable to the system (e.g., observe the fuel temperature). In this way, the ICAI program provides the student with examples of acceptable task performance. The task model is, in a sense, a model of an expert (e.g., Wenger, 1987). The way the task model performs the task is the way the instructional system encourages students to do the same task. That is, the task model represents the knowledge that the student should possess when instruction is complete.

A second characteristic of the task model is that it contains strategies that students are capable of learning. Task models that prescribe strategies that require students to exceed the limitations of short term memory,

for example, are not acceptable since students may not be able to learn how to overcome this limitation. Therefore, this characteristic implies that the task model must take into account human cognitive capabilities and limitations. Although the strategies may not be optimal, the task model represents problem-solving strategies that students can learn.

The third characteristic of the task model is that it organizes knowledge into concepts and metaconcepts. Concepts contain declarative (Anderson, 1976) knowledge or facts. An example of a concept is the set of all observations needed to confirm a particular type of failure (e.g., the set of gauge readings associated with the set of failures resulting in insufficiently heated fuel). Metaconcepts contain procedural knowledge (Anderson, 1976) or a way in which to manipulate concepts (facts) associated with the system. Metaconcepts define the order and type of facts that are gathered. Both concepts and metaconcepts are chosen in a way such that the targeted student population knows all prerequisite information prior to beginning ICAI lessons.

Task Model Structure

To locate failures in a system, troubleshooters iteratively perform two functions: 1) updating the current set of potentially failed components and 2) choosing a test to reduce the size of that set. The proposed architecture uses an operator function model (Mitchell, 1987) to represent these two functions together with concepts and metaconcepts for troubleshooting a complex dynamic system.

Discrete control models (Miller, 1985) and operator function models (OFM) have been successfully used to describe and prescribe operator behavior in a range of complex systems. In particular, OFMs have been used to model, design, and control user interfaces and operator assistants in dynamic systems with advanced levels of automation (Mitchell, 1987; Mitchell and Miller, 1986; Mitchell and Saisi, 1987; Rubin et al., 1988). The OFM is a hierarchic-heterarchic network. The nodes represent operator functions and actions; network arcs represent system events or the results of operator actions that initiate or terminate operator activities. The OFM accounts for the coordination of operator activities and the operator's dynamic focus of attention. As depicted in Figure 4, AIIAB's OFM represents the way an operator might decompose and coordinate troubleshooting actions to identify a system failure. Definitions for each of the nodes depicted in AIIAB's OFM are found in Table 1.

Metaconcepts

AHAB's operator function model for troubleshooting relies on two diagnostic search strategies for choosing tests to perform (Rasmussen, 1986): 1) symptomatic search and 2) topographic search. To perform a symptomatic search, a set of observed symptoms is used as a template or a pattern to be matched in a library of abnormal system conditions. In other words, the present state of the malfunctioning system is compared with various models of a malfunctioning system to determine which failure is causing the symptoms. A topographic search, on the other hand, involves comparing a template of normal system response with the responses observed in the abnormally functioning system. The failure is found by determining the location of a mismatch between the model of the way the system is supposed to work and the observed system state.

Symptomatic Search. Symptomatic knowledge search is an economic failure detection strategy since it can quickly reduce the number of possible failed components with relatively few tests. It is not a general strategy because it requires access to multiple models that represent the way in which the system fails. Thus, a symptomatic search is highly dependent on the availability of data concerning the specific system of interest. Such data are usually gained through experience with a given system. Access to such specific information is the reason symptomatic searches are more economical than topographic searches since specific failure information can lead a troubleshooter to a failed component with relatively few tests. For instance, the presence of a vacuum low alarm will likely lead an expert troubleshooter of a marine powerplant to hypothesize immediately (and correctly) that the vacuum pump has failed. The number of diagnostic tests in this case is greatly reduced because the operator knows that the vacuum low alarm lights up when the vacuum pump fails.

Topographic Search. Topographic knowledge search is more general and less economical than symptomatic knowledge search. It is more general since it is not tied to data representing the great number of possible failed states a particular complex system may possess. The lack of economy also stems from the fact that topographic strategies do not require system information such as the probability of specific parts to fail. Thus, topographic searches are less economical because they do not use all information available to the fullest extent.

A topographic knowledge search compares a normal model of the system and the actual abnormally functioning system to find the location of a failure. Strategies for identifying the location of the mismatch

between the model and the system involve consideration of the system structure and either the magnitude of state variables or the relationships of observed state variable values.

AHAB uses the laws of conservation of mass and conservation of energy as the basis of these relationships. Application of the law of conservation of mass to the example of an incorrectly open valve highlights the symptoms of a low liquid level followed by a high liquid level along the same fluid path and leads a troubleshooter to hypothesize the existence of a leak between the points of those level measurements. Furthermore, since certain components are more likely to cause an observed problem, it is best to test those components first. Since incorrectly opened valves and pumps that are mistakenly turned on are likely causes of leaks along a fluid path, the troubleshooter should check all valves and pumps along that path before checking any other types of component.

Combining Search Techniques. Neither the symptomatic nor the topographic search strategy alone is adequate for all troubleshooting situations. Rather, as an operator solves problems in a complex system, there is a need to switch between the two strategies. AHAB's OFM proposes that a troubleshooter use symptomatic strategies to the extent possible and then switch to a topographic strategy (Hunt and Rouse, 1981).

This use of symptomatic and topographic search results in fewer tests. For instance, if a troubleshooter observes white smoke and abnormal fuel pressure, the set of possibly failed components is quickly reduced to include only those that transport fuel. The troubleshooter knows from the presence of white smoke that there is either not enough fuel or too much air. The abnormal fuel pressure indicates that the problem is probably due to insufficient fuel. The mapping from the white smoke and abnormal fuel pressure symptoms to the identification of components that transport fuel as the set of possible failed components reflects the symptomatic strategy. If the troubleshooter does not know any other symptomatic tests to perform, topographic tests should be used to interpret the pattern of abnormal pressure gauge readings along the fuel path and ultimately to find the failed component.

Concepts

In order to tailor the general operator function model of troubleshooting to fit the specific task of troubleshooting PEQUOD failures, system-specific knowledge (concepts) is embedded within AHAB's operator function model. Although this knowledge affects every node in the OFM network, it primarily concerns the Symptomatic Test and Topographic Test nodes.

Symptomatic Tests. All symptomatic tests for PEQUOD are organized in the feasible set-testing hierarchy shown in Figure 5. The nodes in Figure 5 represent possible sets of components that could have failed given current system knowledge. Along the arcs that connect the nodes in the hierarchy are general tests that can be applied to reduce the size of the current feasible set (i.e., move to a new feasible set that is lower in the hierarchy). Notice that the top-level node in Figure 5 is called All System Parts. Before knowing any symptoms, the feasible set contains all parts in the system.

The hierarchy is similar to the representation of processes in Figure 2 in that both figures show the processes that occur within PEQUOD when it is operating properly (i.e., the square nodes). The feasible set-testing hierarchy, however, also includes the types of failure that occur in PEQUOD (i.e., the oval nodes). Several types of failure are associated with each process. Furthermore, these types of failure are the concepts AHAB teaches using the symptomatic strategies. For instance, concepts for the process of combustion are: 1) incomplete combustion, 2) excessive air and 3) contaminated fuel. The feasible set-testing hierarchy represents the unique symptoms associated with these concepts and the sets of components that should be investigated when those symptoms are observed.

Symptomatic tests for marine powerplant simulator failures were obtained by collecting the pattern of symptoms for each of PEQUOD's twenty-five failures and using these patterns as transition functions among the hierarchical arrangement of feasible sets represented by the feasible set-testing hierarchy. Since symptoms propagate through the system, time is included in this hierarchy by placing tests for symptoms that occur early in the progress of a failure near the top of the hierarchy. Symptoms along the arcs between feasible sets are general, but can be mapped to specific gauge readings.

The first step in using the feasible set-testing hierarchy is to find the current feasible set of possible failed components in the hierarchy. Tests to perform next are found along the arcs descending away from the current feasible set node. Results of those tests are used to update the feasible set.

In most cases, the tests shown in the feasible set-testing hierarchy are straightforward to perform. There are, however, some constraints for the tests. The first constraint has to do with the order in which tests must be performed. Specifically, tests in the feasible set-testing hierarchy must be made in a top-down manner. The reason is that the tests are conditional on the current feasible set. For instance, if the only symptom is an abnormal Water level, it is not clear whether the failure has to do with a Water leak (Improper Feedwater Transport) or a Steam leak (Improper Steam Transport). By performing the tests in the specified order, this question can be resolved.

The second constraint is related to the types of test associated with the current feasible set. When the available tests each represent a yes/no decision concerning a single state variable (e.g., if the feasible set consists of components involved in the Steam Generation process and the test is to check for the presence or absence of smoke), then the choice of a test is easy since only one test (e.g., test for smoke) can be made. If, on the other hand, there are multiple tests given a feasible set node (e.g., when the feasible set node contains parts having to do with Incomplete Combustion and the available tests are to check Boiler Air pressure and temperature, Fuel temperature, or Fuel pressure), then all symptomatic tests leading from the current feasible set node must be performed until one of the tests is satisfied.

The final constraint on using the feasible set-testing hierarchy has to do with the time at which tests may be made. Some tests in the feasible set-testing hierarchy are only appropriate after a certain interval of time has past since the failure. The time intervals during which symptoms may be observed are indicated by the numbers in parentheses in Figure 5. For example, if the feasible set is All System Parts, one of the tests that must be made is to check Superheated Steam pressure. The numbers in parentheses show that if an abnormal Superheated Steam pressure reading related to a problem with the Steam Use process occurs, it will do so between times 0 and 1. Furthermore, since an asterisk (*) follows the number 1, time interval 1 is the last time interval an abnormal Superheated Steam pressure may be observed before there is a conflict with another type of failure (e.g., Steam Generation problems might also cause abnormal Superheated Steam pressures). A normal gauge reading, however, has no time limit.

Topographic Tests. Topographic tests are used to locate failures along a path of components that are related by a common fluid flowing through them (e.g., all the components with water flowing through them). The term topographic test is another way of describing the strategy of looking upstream and downstream from an abnormal gauge reading to find the failure. Topographic tests are based on hypotheses concerning the type of failure PEQUOD is believed to be experiencing. These hypotheses specify patterns to match to confirm or reject a particular type of failure.

Four general types of failure occur in PEQUOD: 1) block in the path, 2) leak along a path, 3) leak beyond the path, and 4) heating problem. A leak along a path occurs, for instance, when a valve is opened but it should be closed. A leak beyond a path happens when, for example, there is a hole in a tank. Each type of failure corresponds to a hypothesis a student may entertain while troubleshooting. All hypotheses and the pattern of gauge readings that must be found in order to confirm each hypothesis are listed in Table 2.

Topographic strategy concepts are associated with combinations of paths (i.e., the lowest level processes in the feasible set-testing hierarchy) and hypotheses. For instance, there are four topographic concepts related to the Water path, one for each of the possible hypotheses for that path. Some paths have fewer concepts associated with them since they have fewer associated hypotheses. For instance, any path in which the fluid is not heated (e.g., Vacuum Air) does not have the associated heating hypothesis or concept.

Notice in Table 2 that the pattern of gauge readings that supports a hypothesis depends on the state of the fluid and on whether or not there is an infinite source (e.g., Atmosphere, which supplies air to components in the Boiler Air path, and the Sea, which supplies Saltwater to components along the Saltwater path) along a fluid path. For instance, if a student is looking for a leak in the Water path, the student should check Water levels, whereas if the student suspects a block in a Steam path, the student should check Steam pressures. The presence or absence of an infinite source is important only when looking for a block along a path. When there is an infinite source in the path, all pressures along the path are low. Blocked paths without infinite sources have pressure or level build-ups behind the block and pressure or level depressions in front of the block.

There are three steps involved in using the knowledge contained in Table 2 to determine which topographic test to perform at a given time:

1. Find a nonnormal gauge reading.

2. Choose a test that will confirm or reject a reasonable hypothesis.
3. When no more gauge readings are available to decrease the size of the feasible set, choose a component in the feasible set and perform a test diagnosis on that component.

Unlike symptomatic tests, topographic tests are applicable at any time interval. If a time limit for a symptomatic test is missed the size of the current feasible set cannot be reduced by using tests in the feasible set-testing hierarchy. In this case the size of the feasible set can still be reduced by using topographic tests. The most economical strategy, however, is to eliminate as many feasible paths as possible by using symptomatic tests and then to apply topographic tests to those few paths.

Specification of the task model is the first step in building an adaptive computer-assisted instruction program. With a functioning task model, the instructional program can represent the prescribed method of performing the task. The task model is then used to develop directed problem-solving aids that convey to students this prescribed method.

Directed Problem-Solving Aids

AHAB's three directed problem solving aids assist in the troubleshooter's two main functions of updating the current feasible set and determining the next test to perform. The Feasible Set aid assists the student in updating the feasible set. The Symptoms-to-Find and Hypotheses-to-Test aids help identify useful symptomatic and topographic tests based on the current feasible set.

Feasible Set. Students use the Feasible Set aid to see the currently feasible paths and components. When AHAB's command, "Feasible Set", is chosen by the student, a list of currently feasible paths is displayed. When a feasible path name is chosen by the student from this list, feasible components along the chosen path and on the currently displayed schematic diagram are marked with a dark spot. Figure 6 shows feasible components along the Fuel path in the Fuel Subsystem.

Symptoms-to-Find. AHAB's Symptoms-to-Find feature shows the tests from the feasible set-testing hierarchy that are applicable, given the current feasible set. When the command "Symptoms-to-Find" is chosen by the student, both the current feasible set and possible next feasible sets are shown, as in Figure 7. The current feasible set is at the left side of Figure 7 and the possible next feasible sets are on the right. From

Figure 7, it is seen that the current feasible set is Incomplete Combustion and the possible next feasible sets are Insufficiently Heated Fuel, Insufficient Air or Air Heating, and Improperly Atomized Fuel.

Choosing one of the next feasible sets results in the display of a list of symptomatic tests to make to determine whether or not the failure is in that next feasible set. For instance, if the current feasible set is Incomplete Combustion and Insufficiently Heated Fuel is chosen, the information in Figure 8 is displayed. Figure 8 summarizes the general tests that must be made in order to verify that Insufficiently Heated Fuel is the next feasible set. It also indicates the subsystems in which to make those tests. Figure 8 shows that in order to discover whether or not the next feasible set is Insufficiently Heated Fuel, a nonnormal Fuel temperature must be found in the Fuel subsystem. Time limits for each test are also provided. Thus, if there is insufficiently heated fuel, a Nonnormal Fuel temperature may be observed at time 3 or later.

If a student is unsure which gauges relate to a particular symptom, the student can choose a suggested test that causes gauge markers on the related schematic diagram to be displayed in inverse video if they represent gauges that should be checked. For example, to check Fuel temperature gauges, the student should choose the line "Fuel Temp NonNormal". When it is chosen, the Fuel temperature gauges on the Fuel Schematic are displayed in inverse video.

If the "Symptoms-to-Find" command is chosen but there are no more symptomatic tests to perform, the student is informed of this fact. The student is also instructed to test hypotheses that relate to the topographic search strategy. Assistance in choosing tests applicable to the various hypotheses associated with topographic tests is given by AHAB's Hypotheses-to-Test aid.

Hypotheses to Test. When Hypotheses-to-Test is chosen, the student is shown a list of all the currently feasible paths. For instance, if the current feasible set is Incomplete Combustion, the feasible paths in Figure 9 are displayed. By choosing one of the feasible path names, the student sees the hypotheses that are applicable for that path. For example, if the Fuel path in Figure 9 is chosen, the student receives information similar to that in Figure 10. Figure 10 shows that the hypotheses applicable to the Fuel path are 1) Leak along the path, 2) Leak out of the path, 3) Block in the path and 4) Heating Problem.

In addition to informing the student of applicable hypotheses, the student is also shown the pattern of gauge readings to seek in order to confirm each hypothesis. As with Symptoms-to-Find, the student can see which

gauges apply to a hypothesis by choosing that hypothesis. For example, by choosing the Heating Problem hypothesis in Figure 10, the temperature gauges along the Fuel path are displayed in inverse video on the Fuel schematic diagram. If a hypothesis is chosen for which all the gauges have been read, the student either is given a list of components that are likely to have failed or is instructed to diagnose any feasible component along the path, depending on whether or not there are any components along that path that are of a type that is more likely to have caused the observed symptoms.

Student Model

The function of the student model is to provide the student with feedback derived from a comparison of the student's actions with the prescribed actions of the expert (i.e., the task model). This feedback informs the student which actions are useful and which are not (i.e., which actions are errors). In this way the student receives coaching while interacting with the instructional system.

Student Model Structure

To create a student model, the actions the student makes while interacting with the instructional system are organized in the context of the expert strategy contained within the task model. Feedback presented to the student is based on a comparison of the student and task models. This feedback represents an assessment of the student's level of understanding of the concepts and metaconcepts included in the task model. To make the comparison of the student and task models more straightforward, the structure of the student model is similar to the structure of the task model. Specifically, AIIAB organizes knowledge in the student model into concepts and metaconcepts.

Besides representing correct actions, AIIAB's student model also represents student errors. An error is defined as any student action the task model would not prescribe, given the current system state and operator goals. Interaction with a large-scale, dynamic system presents students with the opportunity for making a multitude of errors. With so many possible errors, there is little chance that they can be pre enumerated. The best way to account for errors, therefore, is to identify general types of error related to important or common difficulties students experience. If, while a student solves a problem, that student's actions do not match those of the expert, the general rules for making errors are used to identify the error. This method of

determining errors may be thought of as a limited bug model (Wenger, 1987) since it does not attempt to determine all errors a student may make.

Errors in AHAB's student model are associated with both concepts and metaconcepts. Errors in a student's understanding of concepts are detected both while the student interacts with additional instructional programs (e.g., online quizzes) and with the simulator. A student might, for instance, associate incorrect symptoms with a particular type of failure in a troubleshooting quiz. Errors in a student's mastery of metaconcepts, on the other hand, are detected while the student performs the task using the simulator since such errors represent incorrect attempts to perform procedures.

Knowledge contained within the student model is updated as the student interacts with the instructional system. Updating of this knowledge is performed by annotating related concepts and metaconcepts with relevant system and student action information each time a student uses or misuses them. There are two primary ways of updating concepts. First, the program notes when concepts and metaconcepts are correctly used. Due to the large number of possible actions in complex, dynamic systems, "lucky guesses" are far rarer than in small, well-defined problems. Thus, no attempt is made to determine whether or not the student has learned the concept. A second type of updating involves student errors. When a student makes an error, the associated concept or metaconcept is annotated with system state and student action information that indicate why the student's action was an error. Once the student model is updated, feedback is presented to the student based on the current state of the student model and on the rule in the instructional module that determines the amount of feedback a student receives.

Errors in the Student Model

To determine which errors to include in the student model, the appropriate aspects of two related error classification methods were integrated with the operator function model of the troubleshooting task. An error classification method for operators who detect, diagnose and compensate for failures in a high fidelity supertanker engine control room was developed by van Eekhout and Rouse (1981). Johnson and Rouse (1982) developed a scheme for classifying errors in the task of diagnosing failures in an aircraft powerplant. Although neither scheme is exactly suited to the present case, aspects of both were integrated to form the scheme used in the AHAB architecture.

AHAB's error classification scheme has three error categories:

1. **Choice of procedure.** Errors in choice of procedure are made when a student either performs an inappropriate procedure with respect to the current hypothesis or fails to use a systematic procedure in determining the tests to perform.
2. **Execution of procedure.** An error in execution of procedure is committed when a student omits steps, performs steps in an inappropriate order or performs inadvertent actions.
3. **Observation of system state.** Errors in observing the state of the system occur when the student either fails to collect a complete set of information before diagnosing a failure or collects inappropriate information.

Errors concerning both metaconcepts and concepts are placed into one of these three categories. In general, errors associated with metaconcepts are categorized as errors in choice of procedure. Errors in concepts are classified as errors concerning either execution of procedure or observation of system state.

The two errors in choice of procedure are associated with the metaconcepts of coordinating symptomatic and topographic searches and of correctly choosing a test to apply. If a student makes an irrelevant test along a feasible path (e.g., checks Water pressure when only checking Water temperature makes sense), the student may not be using the troubleshooting strategy prescribed by the task model. The student has, therefore, failed to choose the proper symptomatic or topographic troubleshooting strategy. Another way to look at this type of error is that the student's own task model contains an extra node. This extra node contains a strategy that differs from the prescribed symptomatic or topographic strategy. The second choice of procedure error is related to the choice of applied test. Students who do not choose an available symptomatic test over a topographic test have not mastered this metaconcept.

AHAB includes one execution of procedure error. This error is committed when a student makes a test along an infeasible path. A possible cause of this type of error is that the student did not correctly update the current feasible set. Such an error may indicate that the student has not mastered the symptomatic and topographic strategies that enable correct updating of the feasible set.

Three errors concerning the observation of system state are included in AHAB. Two of these errors are related to the symptomatic test knowledge in the feasible set-testing hierarchy. Specifically, a student can make tests in an order or at times other than are specified by the hierarchy. The third type of error in observing the system state is related to the topographic knowledge found in the list of hypotheses to test. Students who perform a diagnosis before reading all gauges associated with a currently supported hypothesis make this type of error.

Interactive Performance Feedback Aids

Table 3 summarizes the three error categories and six errors that AHAB's student model tracks while the student solves PEQUOD problems. In addition to errors, the student model also keeps track of correct student actions. Feedback on these errors and actions is provided as the student interacts with PEQUOD. Two forms of feedback are provided. Error feedback is presented to students each time they make an error. Students also can access feedback concerning correctly applied symptomatic and topographic tests as well as a summary of errors at any time during a problem by using AHAB's Clipboard option. These interactive performance feedback aids are discussed below.

While students solve PEQUOD problems, AHAB monitors their actions to see if the actions match the actions AHAB would suggest. If a student makes a test that AHAB would not make, AHAB displays an error message. Each error causes a message to be displayed on the screen. The way a student obtains a summary of errors made during a problem, as well as a summary of all the symptomatic and topographic tests performed is described below.

Tests made in PEQUOD are classified into one of three categories: 1) Symptoms Found, 2) Hypotheses Tested, and 3) Errors. A student can see previous actions in each category by choosing the Clipboard option. When Clipboard is chosen, the student is presented with three options: Symptoms Found, Hypotheses Tested, and Errors. Symptoms Found lists all tests related to the feasible set-testing hierarchy. Hypotheses Tests lists all actions related to the hypotheses shown in Table 2. Errors lists all actions classified into the types shown in Table 3.

An example list of Symptoms Found is shown in Figure 11. Notice that the initial symptom is listed towards the top of the figure. Following the initial symptom is the set of symptomatic tests listed in the order in which they were performed. The list in Figure 11 indicates that presence of smoke (from the initial symptom) reduced the feasible set from All System Parts to Steam Generation and then to Combustion. The result of a test for the color of the smoke was black and this information reduced the feasible set to those parts having to do with Incomplete Combustion. When fuel temperature was found to be low, the feasible set was reduced to include only those parts having to do with Insufficiently Heated Fuel. Since the feasible set-testing hierarchy does not specify any more tests for the Insufficiently Heated Fuel feasible set,

the feasible set ultimately includes only those parts having to do with Fuel Heating. Thus, Symptoms Found provides a summary of all the tests that are related to the feasible set-testing hierarchy.

An example list of Hypotheses Tested is shown in Figure 12. These hypotheses relate to the fuel path. The figure shows that the Heating Problem hypothesis is supported by a low temperature along the fuel path. A hypothesis can only be rejected if results of tests contradict that hypothesis. Thus, even though no abnormal level readings have been found, according to Figure 12, the Block in the Path, Leak Out of the Path, and Leak Beyond the Path hypotheses cannot be rejected.

An example list of errors is shown in Figure 13. Although the list only shows the total number of each of the six types of error made, the student can find out exactly what errors were made in each category by selecting the name of the error. For instance, if the error category "Tests Along Infeasible Paths" is chosen, the information in Figure 14 is displayed.

Summary

Its ability to represent correct actions and errors enables AIIAB's student model to describe student behavior in troubleshooting PEQUOD failures. By including the student model in the instructional program, a comparison of the student and task models is possible. The comparison allows AIIAB to provide the student with interactive performance feedback in the form of error messages and access to the Clipboard. Through the use of the instructional module, results of the comparison are used to make instructional management decisions. A description of the instructional module follows.

Instructional Module

AIIAB's instructional module coordinates information presented by PEQUOD and other instructional media, as well as by the task and student models. Instructional management decisions made by the instructional module concern the instructional medium, curriculum, pace of instruction, amount of interactive feedback a student receives, and the degree of control the student exercises while interacting with AIIAB and PEQUOD. AIIAB's criteria for making these decisions are described below.

Instructional media to which AIIAB has access include PEQUOD, the dynamic simulator, and a quiz program. The quiz program was designed to help students master the information in the feasible set-testing

hierarchy and in the list of hypotheses. Using five templates for symptomatic search questions and two templates for topographic test questions, the quiz program generates multiple choice questions and a list of possible answers. Templates generate questions that drill students in areas related to AHAB's error categories. For instance, one template generates questions related to the time a particular test can be made. This type of question attempts to help students avoid making errors of conducting symptomatic tests at the wrong time. Moreover, each set of questions is related to a particular concept in the curriculum the student is currently studying.

The basic curriculum specified by AHAB is organized around concepts contained in the feasible set-testing hierarchy. Lessons are ordered from easy to difficult (i.e., lessons of gradually increasing grain size). Concepts associated with Steam Condensation are presented first, followed by Steam Generation and Steam Use concepts respectively. Related concepts within these processes are taught sequentially. For example, all three concepts related to combustion (i.e., Insufficiently Heated Fuel, Insufficient Air or Air Heating, and Improperly Atomized Fuel) are taught together. Recall that hypotheses also represent concepts. Although hypothesis concepts are not included in the initial curriculum specification, if AHAB's instructional module determines that the student has made too many errors concerning a particular hypothesis, that hypothesis concept is reviewed.

Although the general sequence of the lessons is defined by the curriculum, the pace at which students progress through lessons depends on the number of errors they make on previous lessons. If the number of errors made on a given lesson exceeds a specified threshold value, that lesson is reviewed. Students who have more concepts to review see new lessons at a slower pace because they spend more time reviewing previous lessons than students who do not need to review lessons. If there are no lessons to be reviewed, a student is presented with a new topic. When there are lessons to review, however, new and review lessons are alternated. After presentation of a concept, students are questioned about that concept via the quiz program. Following the quiz, students are presented with a number of PEQUOD problems to solve related to the concept.

The amount and timing of instruction presented during a lesson is also determined by the instructional module. Aids available during instruction may not be available after training. To prepare the student for performing the task in a more realistic environment, access to the instructional aids provided by the task and student models is withheld during part of the lesson. In this way, students are less likely to become

dependent on the aid and the transfer of training from the instructional system to the system on which the student will perform the real task will be enhanced.

When the aids are allowed, students receive both the student-initiated and system-initiated feedback. Student-initiated aids include the Feasible Set, Symptoms to Find, Hypotheses to Test, and Clip Board. System-initiated aids include immediate error feedback and the end of problem review. In addition to these system-initiated aids, students who exceed the threshold value for metaconcept errors automatically receive brief tutorial feedback following the problems.

The amount of student control allowed is another decision made by the instructional module. This decision involves the amount of control a student can exercise in accessing directed problem-solving aids and interactive performance feedback. During the portion of the lesson when aids are allowed, the student decides which student-initiated aids to access and when to use them. System-initiated aids are presented automatically. Thus, the student exercises limited control over the instructional interaction.

Knowledge Representation

The AIIAB architecture proposes several requirements for knowledge representation. They include general requirements to represent knowledge from multiple viewpoints, to represent knowledge at multiple levels, and to represent knowledge at the correct grain size. By representing knowledge from multiple viewpoints, directed problem-solving aids have multiple strategies to use in prescribing actions. Furthermore, students' actions are explained in terms of multiple strategies, instead of being forced into only one. Representing knowledge at multiple levels separates concepts from procedures and enables instruction to be tailored to specific aspects of student errors. Finally, representing knowledge at the correct grain size ensures that students are not expected to learn concepts that are too complex or too trivial and also affects the order in which topics are presented.

AIIAB uses frames (Minsky, 1975) and production rules (Newell and Simon, 1972) to represent concepts and procedures, respectively. These specific knowledge requirements are discussed in more detail below. AIIAB requires the task and student models to have a similar structure. In addition, the instructional module must associate with each concept a lesson plan related to that concept. Given these requirements, frames are a convenient mode of knowledge representation and organization. Each concept frame has slots

devoted to the task model, student model, and instructional module. Task model slots contain specific concept information (e.g., a list of related subtasks or related symptoms) and information concerning the relationship of a concept to other concepts. Student model slots contain information about correct student actions and about errors associated with that concept. Instructional module slots contain information related to reading assignments, quizzes, and simulator situations or problems.

Production rules specify the ways in which declarative knowledge contained in the concept frames is manipulated to solve problems. In the task and student models such rules pair specific system conditions with appropriate actions. Production rules also make instruction management decisions using knowledge contained in the concept frames.

Both frames and production rules are important structures in the instructional system. Production rules contain knowledge used in the task and student models. Frames contain knowledge for all three elements of the computer teacher. Each type of knowledge is used to support the functions of providing directed problem-solving aids, interactive performance feedback, and instructional management decisions.

Experimental Evaluation

An experiment was conducted to determine the feasibility and utility of the AIIAB architecture. The experiment addressed two questions:

1. Does the proposed architecture produce an instructional system that yields better troubleshooters than an instructional system with only simulator practice?
2. Is adaptive training necessary, or is it equally effective to provide students with offline aids that present the same normative approach to problem solving?

Both questions address the issue of whether or not the effort of developing an adaptive computer-assisted instruction program is worthwhile. The extra costs associated with developing such programs are difficult to justify if adaptive computer-assisted instruction programs do not exhibit some performance advantage. The answer to the first question affects the decision of whether or not to concentrate on developing an instructional program in conjunction with a simulator. The answer to the second question impacts the decision of whether or not to develop online or offline instructional materials.

With respect to the second question, it should be noted that, although a measure of the **overall effectiveness** of adaptive training can be tested, the experiment does not allow testing of the relative effectiveness of individual components of the instructional program's adaptive capabilities (i.e., the directed problem-solving aid, interactive performance feedback, or adaptive instructional management). Conclusions may, therefore, be made concerning only the overall effectiveness of the adaptive training architecture.

The experimental evaluation designed to answer the above two questions involved two phases: 1) training and 2) testing. During the training phase, students were exposed to one of three instructional methods. In the testing phase, student performance was measured as all students solved the same set of unaided PEQUOD problems. The problem set used during testing included some problems that were seen in training and some problems that subjects had never seen before. The experimental variables, experimental procedure, results of the pilot study, and experiment are described below.

Experimental Variables

The experimental evaluation had two independent variables and four dependent variables. The independent variables were training condition and whether or not the problem had been previously seen (seen status).

Four dependent variables related to AIIAB's prescribed method of troubleshooting were used to assess troubleshooting performance.

Independent Variables

The experimental variable of primary interest was training condition. Three training conditions were considered: unaided simulator practice, non-adaptive (or offline) aiding, and adaptive (or online) aiding. As depicted in Table 4, each of these training conditions required students to solve simulated problems via PEQUOD. Table 4 also shows the various types of aiding associated with each training condition.

All students received simulator practice. Students in the unaided group received offline written material describing processes in a steam powerplant. They did not receive a normative description of the ways PEQUOD may fail and the associated symptomatic and topographic tests to perform. Students in the unaided group solved PEQUOD problems for ten training sessions or until they had solved each training problem at least ten times. The number of problems varied for each student. If a student "timed-out" (i.e., reached a ten minute time limit) on every problem, that student would see fifty problems. Theoretically, therefore, the minimum number of problems seen by any student in the unaided condition was fifty. Problems were presented in random order.

The second training condition was non-adaptive since it used offline aids that remained constant during the training. Students in this condition received offline written material describing processes in a steam powerplant as well as the feasible set-testing hierarchy, and a list of hypotheses to test. Before each PEQUOD problem, students completed an online quiz on symptomatic and topographic tests related to the problem. Students solved problems for ten training sessions. Because of the amount of time spent taking quizzes, students exposed to this training condition saw fewer problems than students in the unaided training group. Therefore, the minimum number of problems offline aiding students saw was some number less than fifty. As in the unaided condition, problems were presented in random order.

Students receiving the third type of training had access to all AIIAB aiding options, including the directed problem-solving aid, interactive performance feedback, and adaptive instructional management. These students also had access to the offline aids and online quizzes that students in the offline aiding group received. Students who completed all the available training lessons with sufficiently few errors could finish training

before the end of the tenth training session. The presence of online quizzes and the various aids associated with AIIAB limited the number of problems students in the online condition could solve during the training sessions. The minimum number of problems students in the online aiding group solved, therefore, was less than the number solved by students in the offline group. Problem order during the training phase depended on the recommendation of AIIAB's instructional management module.

The second independent variable was seen status. Seen status had two levels: previously seen and new. A goal of any training system is to prepare students for both those problems covered in the training and new problems. In this experiment, previously seen problems were problems that students solved during training; new problems were problems that students did not see until the testing phase. By including the seen status variable in the design, the transfer from training material to new material could be analyzed. Due to the limited number of problems available in PEQUOD (25), there were three new and seven previously seen problems in the testing phase.

Dependent Variables

Four primary performance measures were considered in the experiment. These measures assess the ability of the student to solve a problem and the strategy used to do so. Each is described below. For a discussion of all of the measures considered in the experiment, see Fath (1987).

1. **Number of testing problems solved.** A measure of a student's overall performance is the number of testing problems solved. Because a time limit of ten minutes is imposed on each PEQUOD problem, it is possible that a student does not solve all problems. If a student times out on a problem, this fact is noted in the student's data file. Likewise, if a student diagnoses the failed component before the ten minute time limit has expired, that student is said to have solved the problem.

2. **Number of actions.** A measure of a student's overall troubleshooting ability is the number of actions the student requires to solve a problem. The total number of actions a student performs to solve a problem is the total number of gauge readings and diagnoses that student makes during the course of a problem, regardless of whether or not some of the actions can be classified as errors. It should be noted that moving between schematic diagrams is not counted as an action because the student does not gain any diagnostic information from these requests.

3. **Number of symptomatic tests.** The number of symptomatic tests a student performs during a given problem is determined by comparing the automatically provided symptom information (i.e., the symptom information presented initially and updated automatically) and the set of gauge readings the student makes with the tests recommended by the feasible set-testing hierarchy. Symptomatic tests may involve no actions or multiple actions. Symptomatic tests involve no actions when relevant symptomatic information is presented automatically in the initial or subsequent symptoms for a problem. In some cases, students have to check several related gauge readings to ensure a normal general symptom (e.g., check Superheated Steam pressure) or they may have to check several unrelated gauge readings associated with an "OR" expression in the feasible set-testing hierarchy (e.g., check Desuperheated Steam temperature OR check for Gas smoke). Thus, a single symptomatic test may also involve reading multiple gauges.

4. **Number of topographic tests.** The number of topographic tests a student makes in the course of solving a problem is computed by comparing each gauge reading the student makes with the set of currently applicable hypotheses. If the gauge reading is applicable to more than one hypothesis, then the number of topographic tests is increased by the number of hypotheses to which the gauge reading applies. Thus, the number of topographic tests can be greater than the number of actions since one gauge reading can be used to test as many as three hypotheses.

The dependent variables chosen are closely related to the troubleshooting strategy defined by the feasible set-testing hierarchy and hypotheses. Students who were not taught the strategy were expected to make more errors and fewer useful tests. A limitation of these dependent measures was that a student may appear to be using the prescribed strategy but in reality might not be. As mentioned previously, however, the chances of a student correctly guessing the solution to each problem in such a complex system are small. Thus, this limitation should not be significant.

Experimental Procedure

The experiment consisted of approximately ten training sessions and up to two testing sessions per subject. Each session lasted approximately 50 minutes. In both the offline and online aiding groups, the aid was withheld during the second half of each training session. In this way, subjects could prepare for performing the task of unaided problem solving, which is the purpose of the training program.

In the testing sessions each subject received ten unaided PEQUOD problems. The problems in the test were presented in random order. After completing the ten test problems, subjects were asked to complete a questionnaire designed to assess the quality of the training experience. Performance data from the testing sessions and responses to the questionnaires comprised the data for this experiment.

Sophomore students were recruited from the Georgia Institute of Technology Navy ROTC program. Subjects had completed three basic Naval Science courses at Georgia Tech and had a small amount of shipboard experience. Twenty-four students served as subjects in the experiment.

Although the subjects were unpaid volunteers, they competed for three $25 prizes. One prize was awarded in each training condition. Prizes were given to the student in each training group who solved all test problems and used the lowest average number of actions.

Results

Three methods were used to analyze the experimental data. First, the effect of training condition on the number of solved test problems is examined. Next, the effects of training condition and whether or not a problem was seen during the training (i.e., seen status) are discussed. Finally, the student questionnaire results are presented.

Number of Solved Problems

Of the 240 test problems attempted (i.e., ten problems for each of twenty-four subjects), one problem had to be omitted from both the unaided and online aiding conditions due to an unavoidable difficulty in the testing mechanism. Table 5 summarizes the number of solved and unsolved problems for each experimental condition.

Table 5 shows that subjects in the offline aiding group solved all the test problems while subjects in the unaided and online aiding groups failed to solve two and eight problems, respectively. The results of the Chi-Square Test of Association (Hopkins and Glass, 1978) show that there is a significant ($p < 0.01$) association between the training condition and the number of problems solved. Thus, according to this analysis, subjects in the online aiding condition solved significantly fewer problems than subjects exposed to the other training conditions.

Closer examination of the data, however, reveals that one subject in the online aiding condition was responsible for five of the unsolved problems in that condition. Since no other subject failed to solve more than one problem, this subject may be considered to be an outlier. When this subject's data are eliminated, results of the Chi Square Test of Association indicate that there is no significant association between the number of problems solved and the training condition.

Subject Actions

Analyses of subject actions were conducted using only those data from test problems that were solved. Thus, the total number of observations for this analysis was 228. A separate analysis was conducted for each of the three performance measures.

Analyses of subject action data were conducted by using the SAS General Linear Model (GLM) procedure to construct analysis of variance tables. Because the experimental design was unbalanced, the Type III sums of squares option was employed. Satterthwaite's approximation (Montgomery, 1984) was used to calculate an approximate F statistic since the SAS sums of squares are not necessarily independent. Tables 6 a, b, and c [2] summarize the results of GLM analyses performed on the subject action and error measures, respectively.

Actions. The performance measure most indicative of overall problem-solving performance is the number of actions a subject used to solve a problem. The GLM analysis of Actions indicates two major results. First, the number of actions depends on whether or not the subject had previously seen a problem. Subjects made fewer actions ($p < 0.01$) to solve problems they had seen before than to solve new problems. This result is indicative of subjects' use of symptomatic search strategies based on familiarity with the observed pattern of symptoms for a problem. For previously seen problems, a subject may recognize the initial symptom or the pattern of observed gauge readings and can solve the problem using fewer actions. If the symptom can be directly mapped to a failed component, the subject is using a symptomatic strategy, even though this particular strategy is not directly represented in the feasible set-testing hierarchy.

The second result is the significant interaction between training condition and seen status. Figure 15 illustrates this interaction. For previously seen problems, there is a significantly increasing trend towards more total actions as more aiding was added to the training. Subjects who received no aiding during training used the fewest actions. Subjects who received offline aiding used more actions to solve problems and subjects who received online aiding used the most actions. This result may be due to the fact that as more aiding was added to the training, subjects saw each problem fewer times before taking the test since more of their

[2] Only those values of p that were less than .10 are included in Tables 6 a through c.

time was spent interacting with the aid[3]. Subjects in the unaided, offline, and online aiding groups saw each problem an average of 7.3, 3.2, and 2.0 times respectively prior to the test. Thus, unaided subjects saw a "seen" problem more than three times as often as did subjects who received online aiding.

More important than performance on previously seen problems is the subjects' performance on new problems. Figure 15 also shows that for new problems, significantly fewer actions are associated with the online aiding condition than with any other condition. One hypothesis supported by this research is that subjects who received online aiding performed better on new problems. This result may be due to their mastery of the troubleshooting techniques presented in the training. Such mastery was promoted by the elements of the training system that allow it to be adaptive. Evidence that subjects in the online aiding group mastered the proposed troubleshooting techniques is also supported by the fact that their performance is more consistent across seen and new problems than performance of subjects in the unaided or offline aiding groups.

Applied Symptomatic Tests

Training condition was the most influential ($p < 0.01$) factor in the analysis of the number of symptomatic tests applied at the correct time and in the correct order as specified by the feasible set-testing hierarchy. Subjects in the unaided group correctly applied the fewest symptomatic tests. This fact is not surprising since subjects in the unaided group did not receive training based on the feasible set-testing hierarchy. It is noteworthy, however, that subjects in the online aiding group correctly applied more symptomatic tests than did subjects in the offline aiding group. This result provides further evidence that subjects in the online training group more fully mastered the prescribed troubleshooting techniques than did subjects in the offline training group. The result may also be a reason for the better performance of the online aiding group in solving new problems.

Applied Topographic Tests

As in the case of the analysis of the total number of actions required to solve a problem, there was a significant main effect of seen status as well as a significant interaction between training condition and seen status

[3] Recall that training time was held constant for all training conditions. As a result, the number of problems subjects saw during training varied.

for applied topographic tests. With respect to the seen status main effect, subjects performed significantly more topographic tests for new problems than when they had seen a problem at least one time before the test ($p < 0.01$). This result is not surprising since subjects who have not previously seen a problem will not be able to take short cuts to find the failed component that familiarity (i.e., symptomatic search) often allows.

Figure 16 illustrates the training condition by seen status interaction. Subjects in the unaided group solving previously seen problems made the fewest topographic tests. Again, this result can be explained by the differing amounts of practice subjects in the three training groups received. Subjects in the unaided and offline aiding groups, while solving new problems, made the most topographic tests. Subjects in the online aiding group, however, performed approximately the same number of topographic tests regardless of whether or not they had seen a problem before. Thus, these subjects used a consistent strategy whether they had previously seen a given test problem or not. This result indicates that subjects in the online aiding condition learned and used topographic tests, while subjects in the other groups did not. Subjects in the unaided group did not learn the topographic strategy through experience. Subjects in the offline aiding group did not learn this strategy as well as subjects that received online aiding. Mastering this strategy helped subjects solve new problems with fewer actions than subjects in the other groups.

Questionnaire Responses

The purpose of the questionnaire was to assess:

1. the extent to which subjects in aided groups used the aid
2. the usefulness of the aid for solving aided problems
3. the extent to which subjects in aided groups used the suggested strategy while solving unaided problems
4. whether subjects in all conditions believed that they were better troubleshooters as a result of the training they received.

Subjects in the offline and online aiding groups received the same questionnaire. Because students in the unaided group were not taught the prescribed troubleshooting strategy, they received a different questionnaire. Since one subject in the aided group did not return his questionnaire, the total number of completed questionnaires was twenty-three.

Analyses of variance performed for questions in each category showed no significant difference between training conditions. The overall response to the training system was positive, regardless of the training condition to which a student was exposed. Important points raised in the questionnaire responses are listed below:

1. Not all subjects in the offline and online aiding groups used the aid. Those who did use it, however, responded positively to it. Some subjects said that they used the aid when they were first learning about the system or when they did not recognize a problem.

2. Subjects differed on the aspects of the aid that made it useful. This result supports the idea of providing multiple aids and allowing subjects to choose which ones to use.

3. Most subjects used the prescribed strategy to solve unaided problems.

4. Subjects in all training conditions were asked to assess the extent to which they felt the training helped them improve their troubleshooting skills. Most students believed that the training system did help them to become better troubleshooters.

Conclusions

Results of the experiment show that, although students in all three training groups solved approximately the same percentage of test problems, their strategies for solving the problems were different Students in the unaided and the online aiding groups developed widely different strategies because, even though students in the unaided group received more practice on the training problems, they did not devise an expert strategy for dealing with new problems. Conversely, because students in the online aiding group spent much of their time learning an expert strategy, they did not receive as much practice on training problems and could not memorize the failed components associated with various symptoms. Therefore, students who received online aiding did not perform as well on previously seen problems as students who received no aiding, but they did perform significantly better on new problems. Furthermore, because they learned a strategy for solving problems, students in the online aiding group performed more consistently for previously seen and new problems than students in the other training groups. Therefore, both practice and the learning of an expert strategy are important for becoming proficient troubleshooters in complex systems.

Because students in the offline aiding group received an intermediate amount of practice on training problems, their performance on previously seen problems lies between the performance of unaided and online aided students. The facts that 1) offline aiding students performed fewer symptomatic tests than online aiding students and 2) the performance of offline aiding students on new problems is comparable to the performance of unaided students on new problems suggest that offline aiding students did not learn the expert troubleshooting strategy as well as online aiding students. This result supports the conclusion that adaptive training enabled students in the online aiding group to learn the expert troubleshooting strategy better. In this case, it was not sufficient to present the instructional material offline. Rather, it was necessary to provide adaptive instructional management, directed problem-solving aids, and interactive performance feedback.

In any instructional system, the opinions of the people who must use the program are important. Student questionnaire responses indicate that students were generally positive about the system. Students used the aid, especially when they were unfamiliar with a problem. Students found the aid useful in solving aided problems. It is noteworthy that students did not agree on the usefulness of the various online aiding options. An important aspect of adaptive training is that it can provide multiple aids and allow the student

to choose when and how to be aided. Students said they used the prescribed strategy to solve unaided problems. Finally, students felt that their experience with the system was helpful in making them better troubleshooters.

The results of this experimental evaluation support the hypothesis that training with an adaptive online system built using the proposed architecture produces better performance than training using simulator practice alone, at least with unfamiliar problems. Such adaptive training, combined with increased simulator-based practice on training problems should result in superior performance for both new and previously seen problems. Furthermore, it is not sufficient to develop an expert strategy and present it to students using offline materials. The training is most effective if it adapts to individual student needs.

References

1. Anderson, J. R. *Language, Memory, and Thought.* Hillsdale, NJ: Lawrence Erlbaum, 1976.

2. Burton, R. R., and Brown, J. S. An investigation of computer coaching for informal learning activities. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, New York: Academic Press, 1982.

3. Clancey, W. J. *Knowledge-based Tutoring: The GUIDON Program,* Cambridge, MA: MIT Press, 1987.

4. Emmett, A. Discovering a new way to learn. *Personal Computing,* pp. 52-62+, January 1984.

5. Fath, J. L. An architecture for adaptive computer-assisted instruction programs for complex, dynamic systems. Georgia Institute of Technology, CMMSR Technical Report 87-3, 1987.

6. Govindaraj, T. Qualitative approximation methodology for modeling and simulation of large dynamic systems: Application to a marine powerplant. *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-17, no. 6, pp. 937-955, November/December 1987.

7. Hollan, J. D., Hutchins, E. L., and Weitzman, L. STEAMER: An interactive inspectable simulation-based training system. *The AI Magazine* pp. 15-27, Summer 1984.

8. Hopkins, K. D. and Glass, G. V. *Basic Statistics for the Behavioral Sciences.* Englewood Cliffs, NJ: Prentice-Hall, 1978.

9. Hunt, R. M. and Rouse, W. B. Problem solving skills of maintenance trainees in diagnosing faults in simulated powerplants. *Human Factors*, vol. 23, no. 3, pp. 317-328, 1981.

10. Johnson, W. B. and Rouse, W. B. Analysis and classification of human errors in troubleshooting live aircraft power plants. *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-12, no. 3, pp. 389-393, May/June 1982.

11. Kratt, F. M. and Govindaraj, T. Multi-fidelity simulation training for marine engineers. Marine Safety International, New York, Internal rep. no., April 17, 1984.

12. Miller, R. A. A systems approach to modeling discrete control performance, in W. B. Rouse (Ed.), *Advances in Man-Machine Systems Research,* Volume II, Greenwich, CT: JAI, 1985, pp. 177-248.

13. Minsky, M. A framework for representing knowledge. In P. Winston (Ed.), *The Psychology of Computer Vision.* New York: McGraw-Hill, 1975.

14. Mitchell, C. M. GT-MSOCC: A research domain for modeling human-computer interaction and aiding decision making in supervisory control systems, *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-17, no. 4, pp. 553-570, July 1987.

15. Mitchell, C. M. and Miller, R. A. A discrete control model of operator function: A methodology for display design, *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-16, no. 3, pp. 353-369, May/June 1986.

16. Mitchell, C. M. and Saisi, D. L. Use of model-based qualitative icons and adaptive windows in workstations for supervisory control systems. *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-17, no. 4, pp. 573-593, July/August 1987.

17. Montgomery, D. C. *Design and Analysis of Experiments.* New York: John Wiley & Sons, 1984.

18. Newell, A. and Simon, H. A. *Human Problem Solving*, Englewood Cliffs, NJ: Prentice Hall, 1972.

19. Rasmussen, J. *Information Processing and Human-Machine Interaction,* North-Holland, New York, 1986.

20. Stevens, A., Roberts, B., and Stead, L. The use of a sophisticated graphics interface in computer-assisted instruction. *IEEE Computer Graphics and Applications*, pp. 25-31, March/April 1983.

21. Towne, D. M. The generalized maintenance trainer: evolution and revolution. In W. B. Rouse (Ed.), *Advances in Man-Machine Systems Research,* Volume III. Greenwich, CT: JAI Press, Inc., 1986.

22. van Eekhout, J. M. and Rouse, W. B. Human errors in detection, diagnosis, and compensation for failures in the engine control room of a supertanker. *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-11, no. 12, pp. 813-816, December 1981.

23. Wenger, E. *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge,* Los Altos, CA: Morgan Kaufman Publishers, Inc., 1987.

24. Williams, M., Hollan, J. and Stevens, A. An overview of STEAMER: An advanced computer-assisted instruction system for propulsion engineering. *Behavior Research Methods & Instrumentation*, vol 13, no. 2, pp. 85-90, 1981.

Tables

Table 1. Definitions of Constraint Network Nodes

- Iteration - Iteration reflects the number of complete passes through the model that have been made so far. For each iteration, a diagnosis or a testing action is performed.
- Symptoms - At the beginning of a problem (and sometimes during a problem) the troubleshooter is presented with symptoms of the failure. The symptoms translate to gauge readings. Each symptom and all gauge readings the troubleshooter observes are added to the list of symptoms.
- Initial Feasible Set - The term "initial" in this context refers to the segment of the current iteration prior to updating the feasible set, which occurs in the Current Feasible Set node.
- Current Feasible Set - The feasible set that reflects the application of all observed symptoms to the initial feasible set is the Current Feasible Set.
- Time - Time is measured with respect to the simulator dynamics. It is an indication of the number of time intervals that have elapsed since symptoms of the failure first appeared. The current time may be read directly from PEQUOD's clock.
- Symptomatic Test - A test that is determined by using a symptomatic search strategy is a symptomatic test.
- Topographic Test - A topographic test is a test determined through use of a topographic search strategy.
- Diagnosis - A diagnosis is performed when there is only one component in the feasible set. When this is the case, the troubleshooter can be certain that the remaining feasible component is the failed component.
- Applied Test - The applied test is the symptomatic or topographic test that is performed in the current iteration. Preference is given to symptomatic tests since they tend to be more economical.

Table 2. Hypotheses

Description	Fluid State	Infinite Source?	Pattern of Gauge Readings that Supports the Hypothesis		
Block	Liquid	No	Level	High	Low
	Gas	No	Pressure	High	Low
		Yes	Pressure	Low	Low
Leak along the path	Liquid		Level	Low	High
	Gas		Pressure	High	High
Leak beyond the path	Liquid		Level	Low	Low
	Gas		Pressure	Low	Low
Heating problem	Liquid		Temperature		Low
	Gas		Temperature	Low	Low

Table 3. AHAB's Error Classification Scheme

Category	Errors
Choice of procedure	1. Irrelevant test along a feasible path 2. Hypothesis tested before all symptomatic tests are performed
Execution of procedure	1. Test along an infeasible path
Observation of system state	1. Symptomatic test performed in the wrong order 2. Symptomatic test performed at the wrong time 3. Diagnosis made before all relevant gauges are read

Table 4. Training Conditions

Type of Aid	Unaided	Offline Aids	Online Aids
Simulated problems via PEQUOD	X	X	X
How the system works	X	X	X
Suggested Tests			
1 Feasible set-testing hierarchy		X	X
2 Hypotheses to test		X	X
Online Quiz		X	X
Directed Problem Solving Aid			X
Interactive Performance Feedback			
1 Online error checking			X
2 Clipboard			X
Adaptive Instructional Management			X

Table 5. Numbers of Solved and Unsolved Test Problems.

	Training Condition		
	Unaided	Offline	Online
Solved	77	80	71
Unsolved	2	0	8

Table 6a. Levels of Significance for GLM Analyses of Actions

Source	DF	Type III SS	MS	MS"	$DF_{MS"}$	F	PR>F
Condition	2	1012	506	796	22	.64	
Seen	1	5380	5380	317	56	16.99	.01
Condition*Seen	2	3073	1536	448	23	3.43	.05
Problem(Seen)	10	4311	431	182	170	2.37	.01
Subject(Condition)	21	18853	880	480	21	1.83	.10
Seen*Subject(Condition)	21	10087	480	182	170	2.64	.01
Error	170	81063	182				

Table 6b. Levels of Significance for GLM Analyses of Symptomatic Tests

Source	DF	Type III SS	MS	MS"	$DF_{MS"}$	F	PR>F
Condition	2	55.40	27.70	2.12	23.6	13.06	.01
Seen	1	0.18	0.18	15.99	11.3	.01	
Condition*Seen	2	4.94	2.47	1.63	24.4	1.52	
Problem(Seen)	10	653.22	65.32	1.03	170.0	63.42	.01
Subject(Condition)	21	47.61	2.27	1.70	21.0	1.34	
Seen*Subject(Condition)	21	35.65	1.70	1.03	170.0	1.65	.05
Error	170	170.70	1.03				

Table 6c. Levels of Significance for GLM Analyses of Topographic Tests

Source	DF	Type III SS	MS	MS"	$DF_{MS"}$	F	PR>F
Condition	2	10094	5047	6442	22.2	.78	
Seen	1	37137	37137	2831	49.8	13.12	.01
Condition*Seen	2	23955	11978	3955	22.8	3.03	.10
Problem(Seen)	10	43286	4329	1423	170.0	3.04	.01
Subject(Condition)	21	149652	7126	4268	21.0	1.67	
Seen*Subject(Condition)	21	89637	4268	1423	170.0	3.00	.01
Error	170	241894	1423				

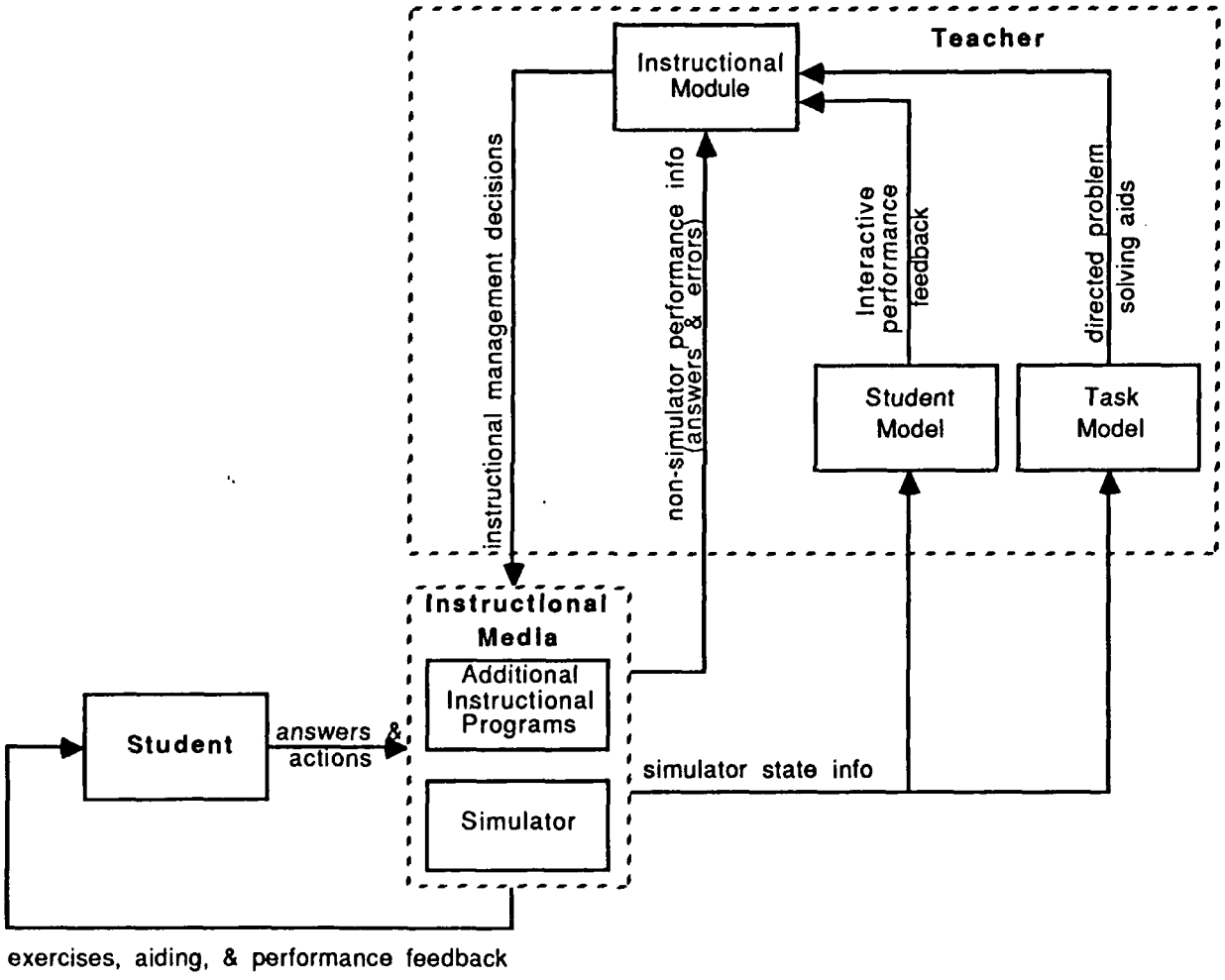


Figure 1. Architecture for an Instructional System

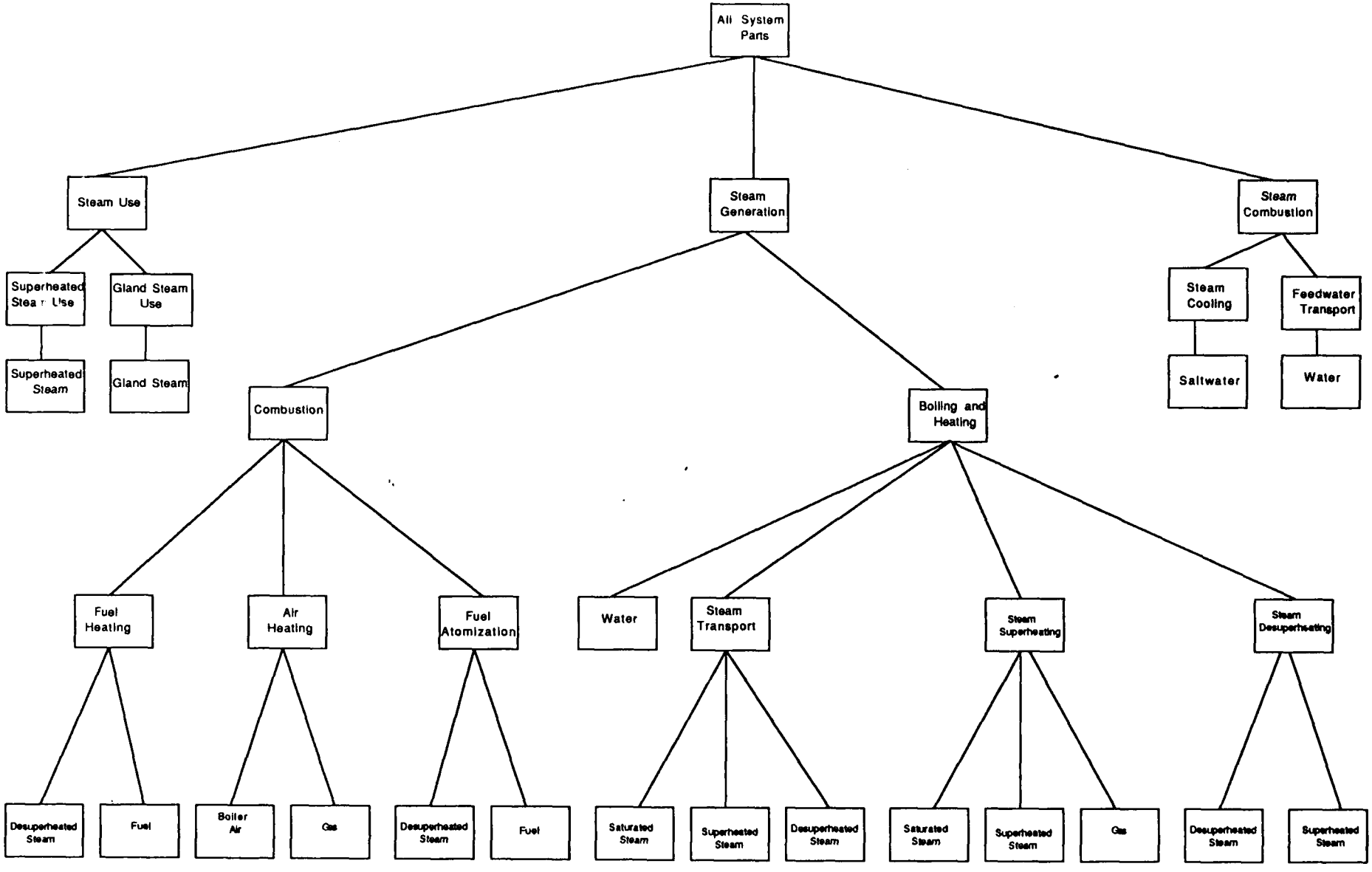


Figure 2 Processes in PEQUOD

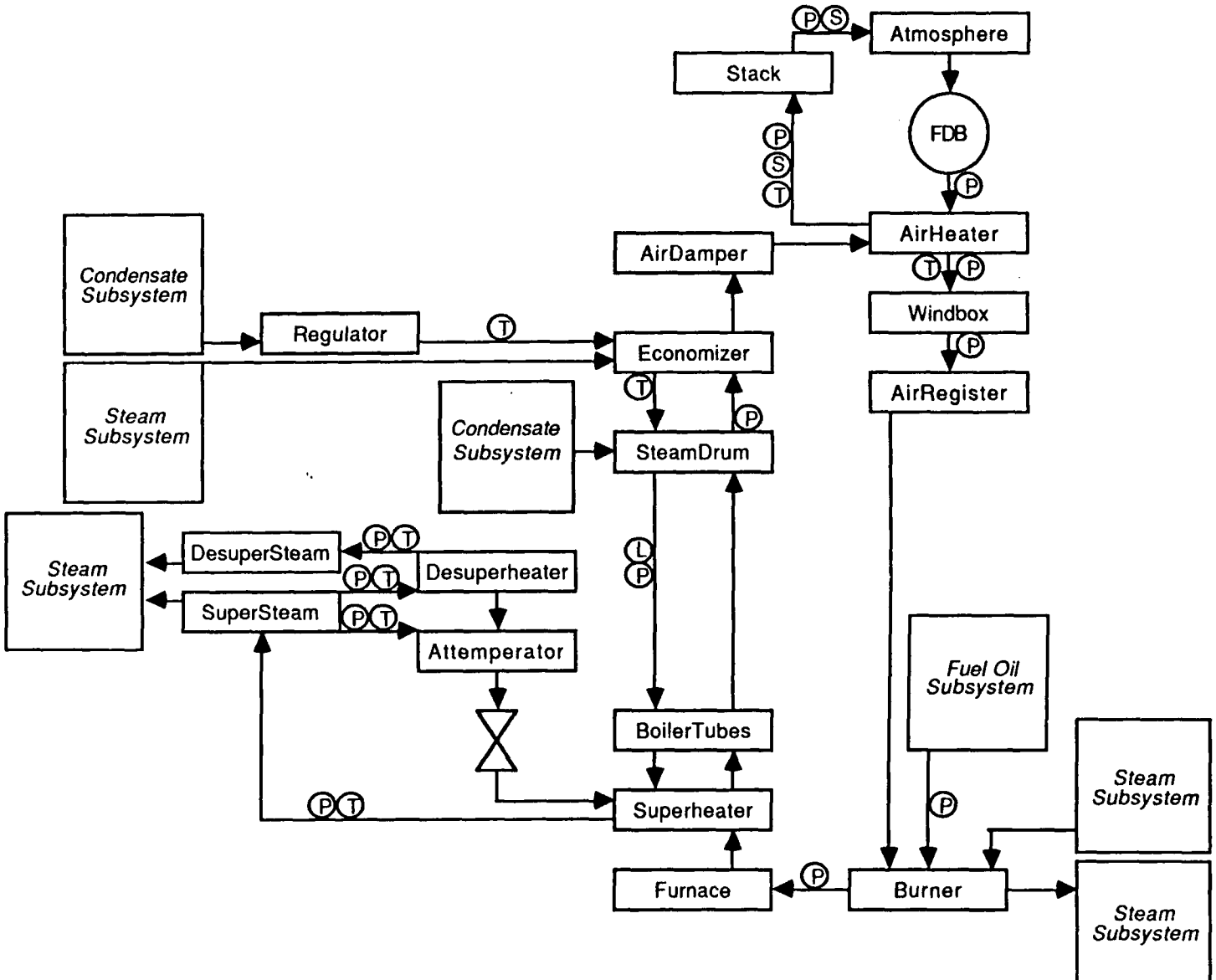
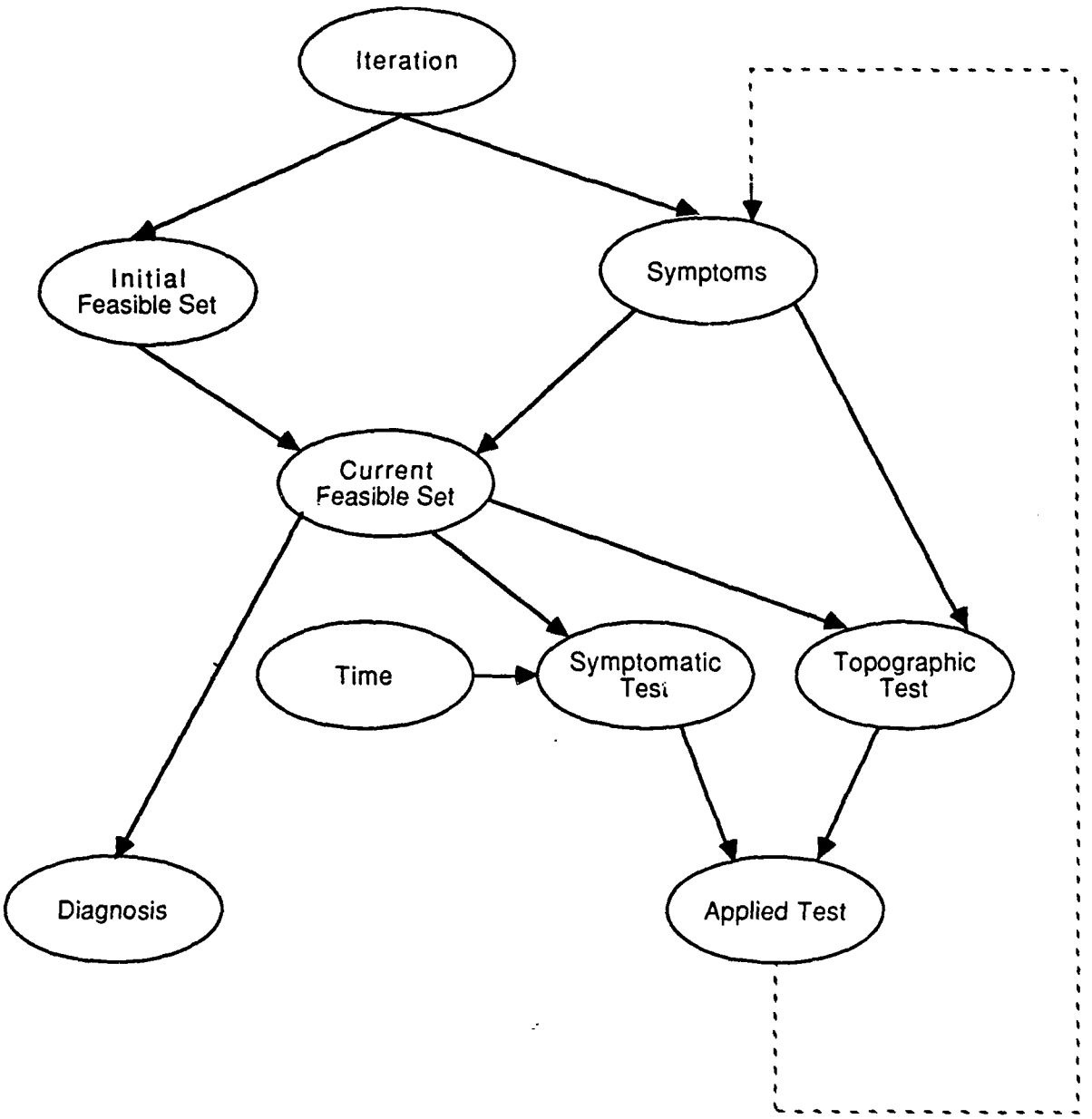


Figure 3. The Boiler Subsystem

Figure 4. AHAB's Operator Function Model



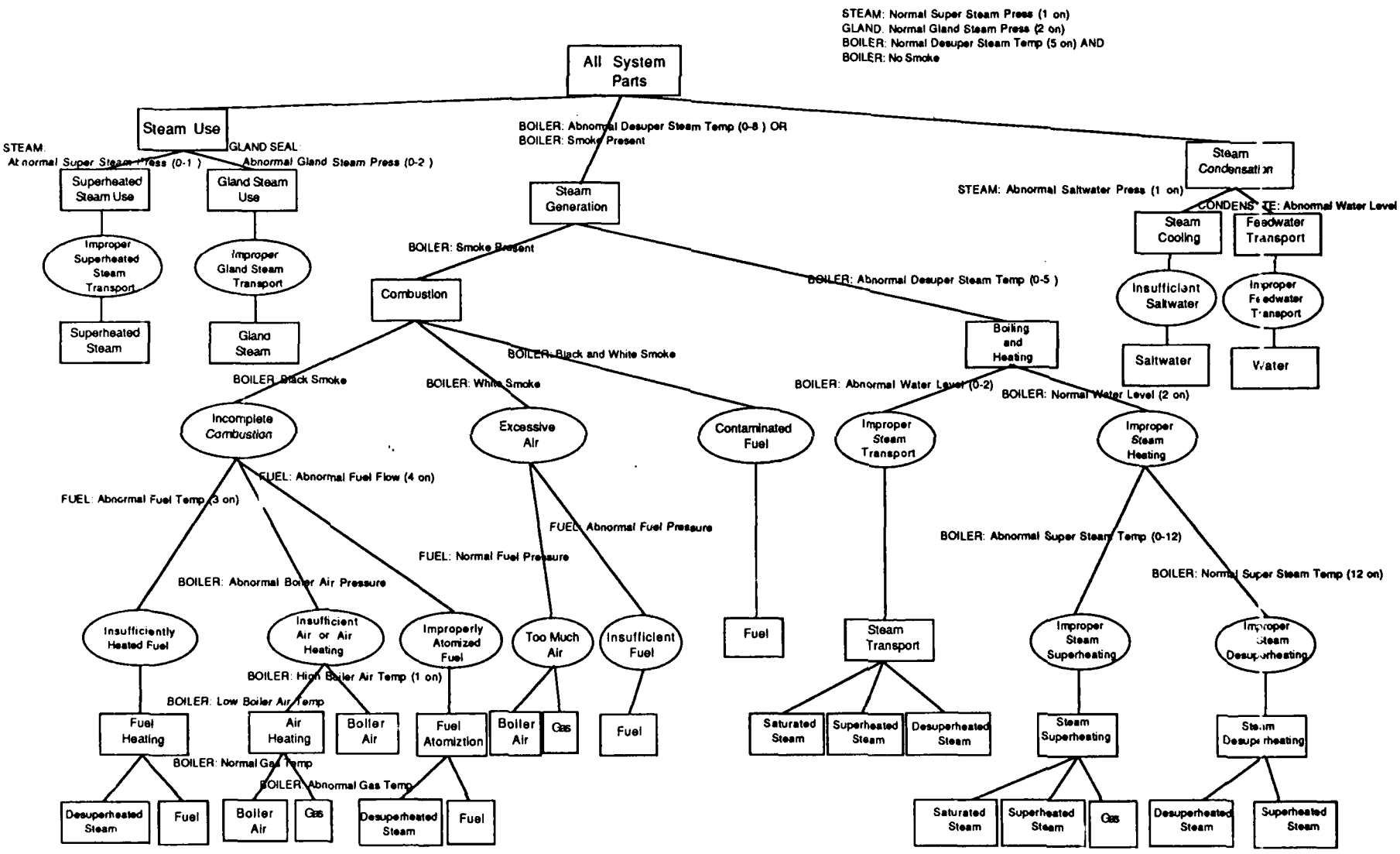


Figure 5 The Feasible Set-Testing Hierarchy

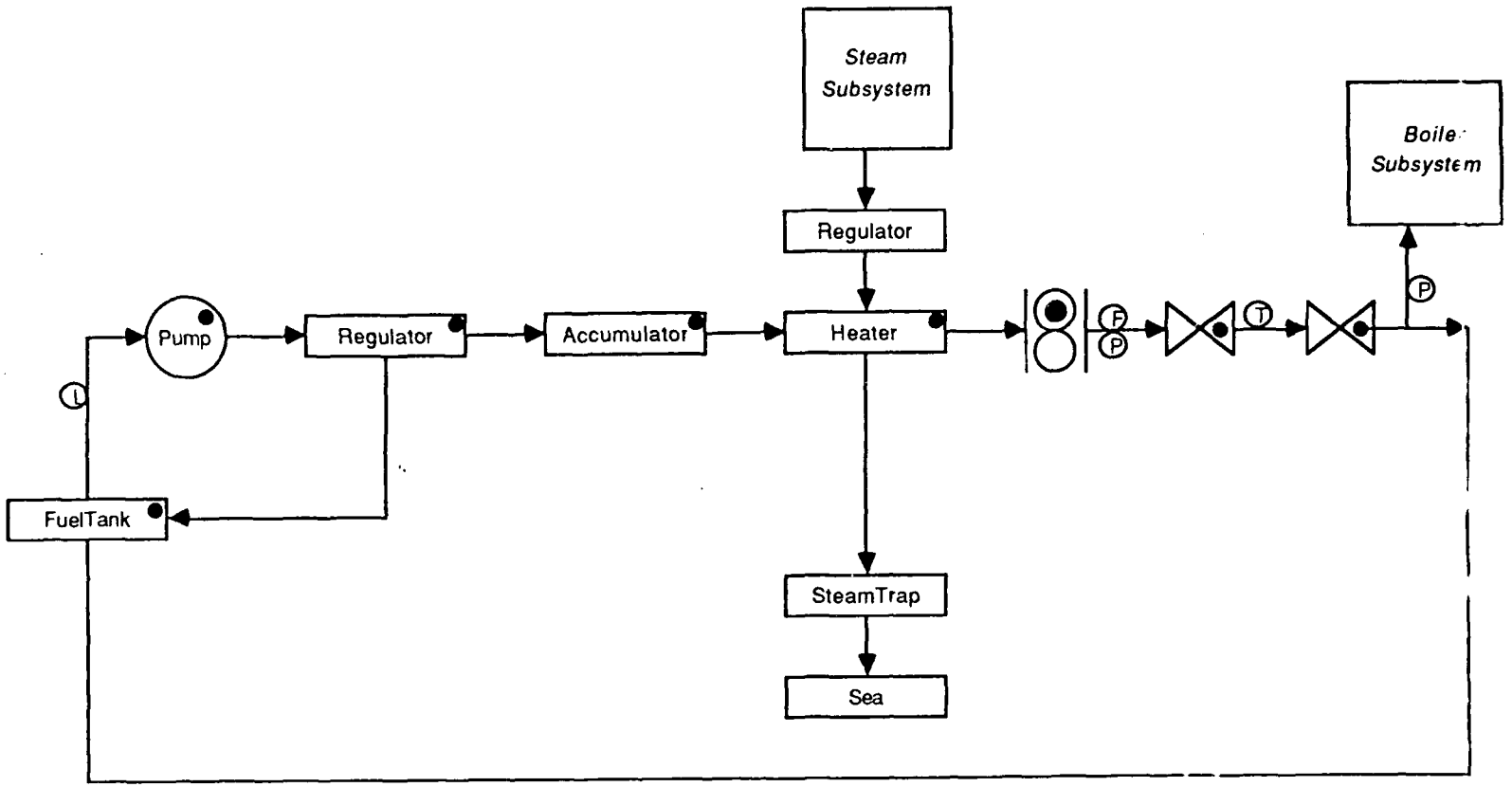


Figure 6. Feasible Components Along the Fuel Path in the Fuel Oil Subsystem

Figure 7. The Current Feasible Set and Three Next Feasible Sets

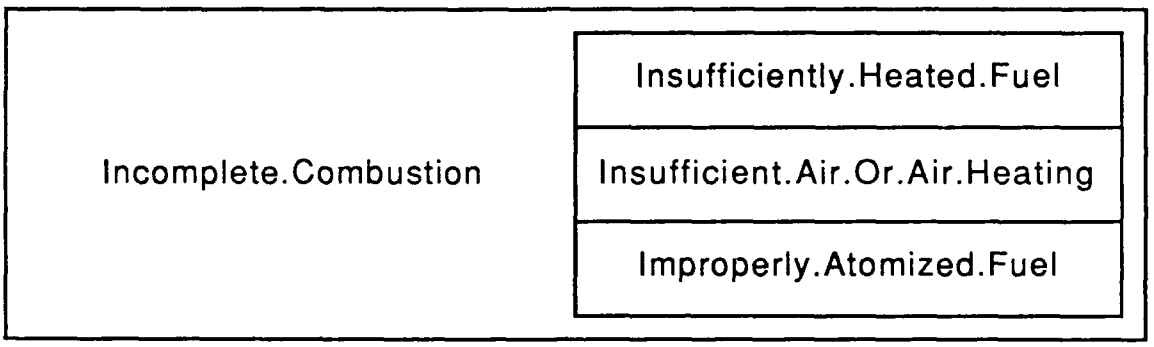


Figure 8. Tests to Perform to Verify the Insufficiently Heated Fuel Feasible Set

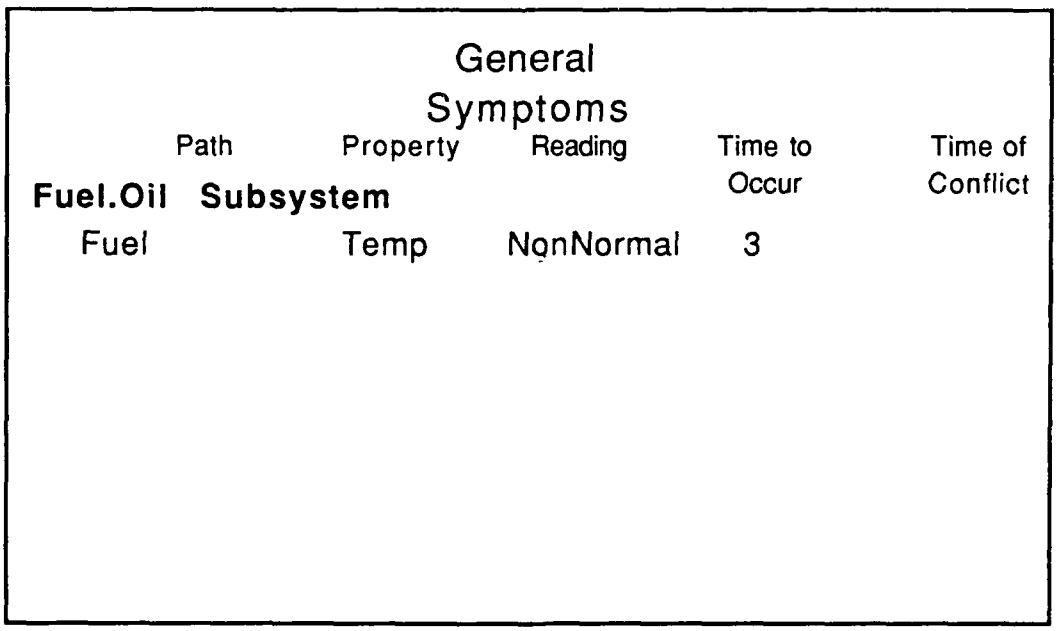


Figure 9. Feasible Paths Related to Hypotheses to Test

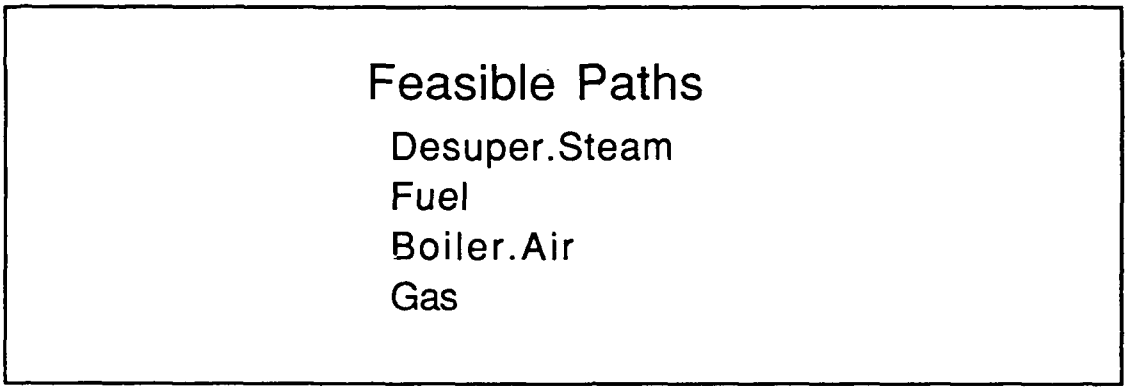
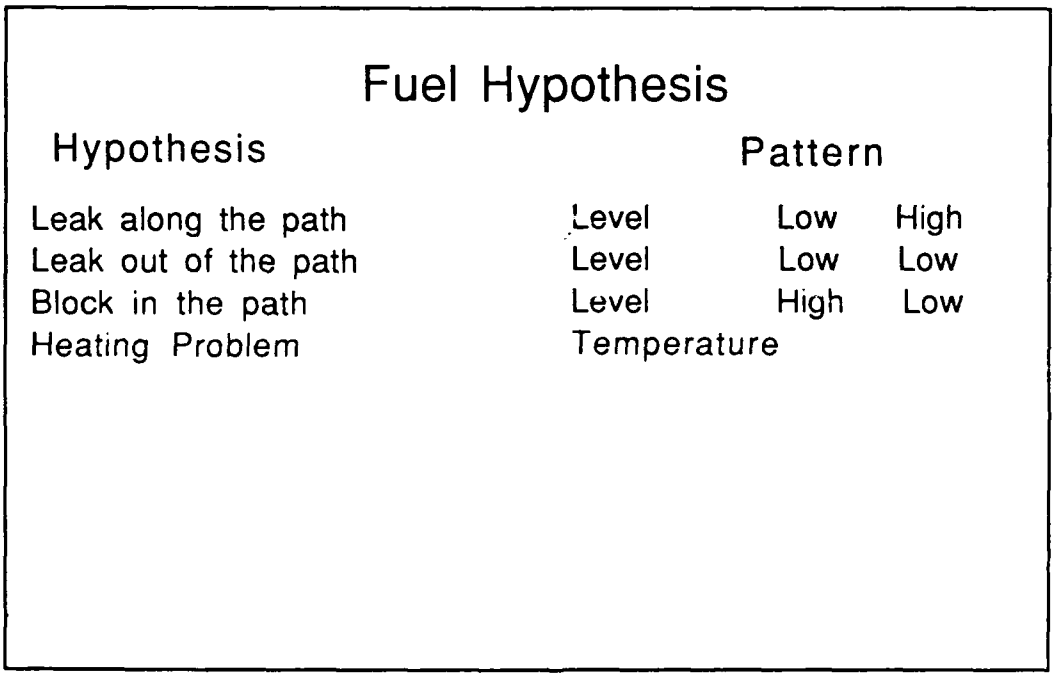


Figure 10. Hypothesis Related to the Fuel Path



Clip Board

Symptoms Found | Hypothesis Tested | Errors Made

Initial Symptom:

The boiler is smoking

Feasible Set Before Test	Applied Test Path	Property	Reading	Iteration
All.System.Parts	Gas	Smoke	Present	0
Steam.Generation	Gas	Smoke	Present	0
Combustion	Gas	Smoke	Black	2
Incomplete.Combustion	Fuel	Temperature	Low	3
Insufficiently.Heated.Fuel	** No Test was necessary **			
Fuel.Heating	*** There are no more symptoms to find ***			

Figure 11. Example Clipboard List of Symptoms Found

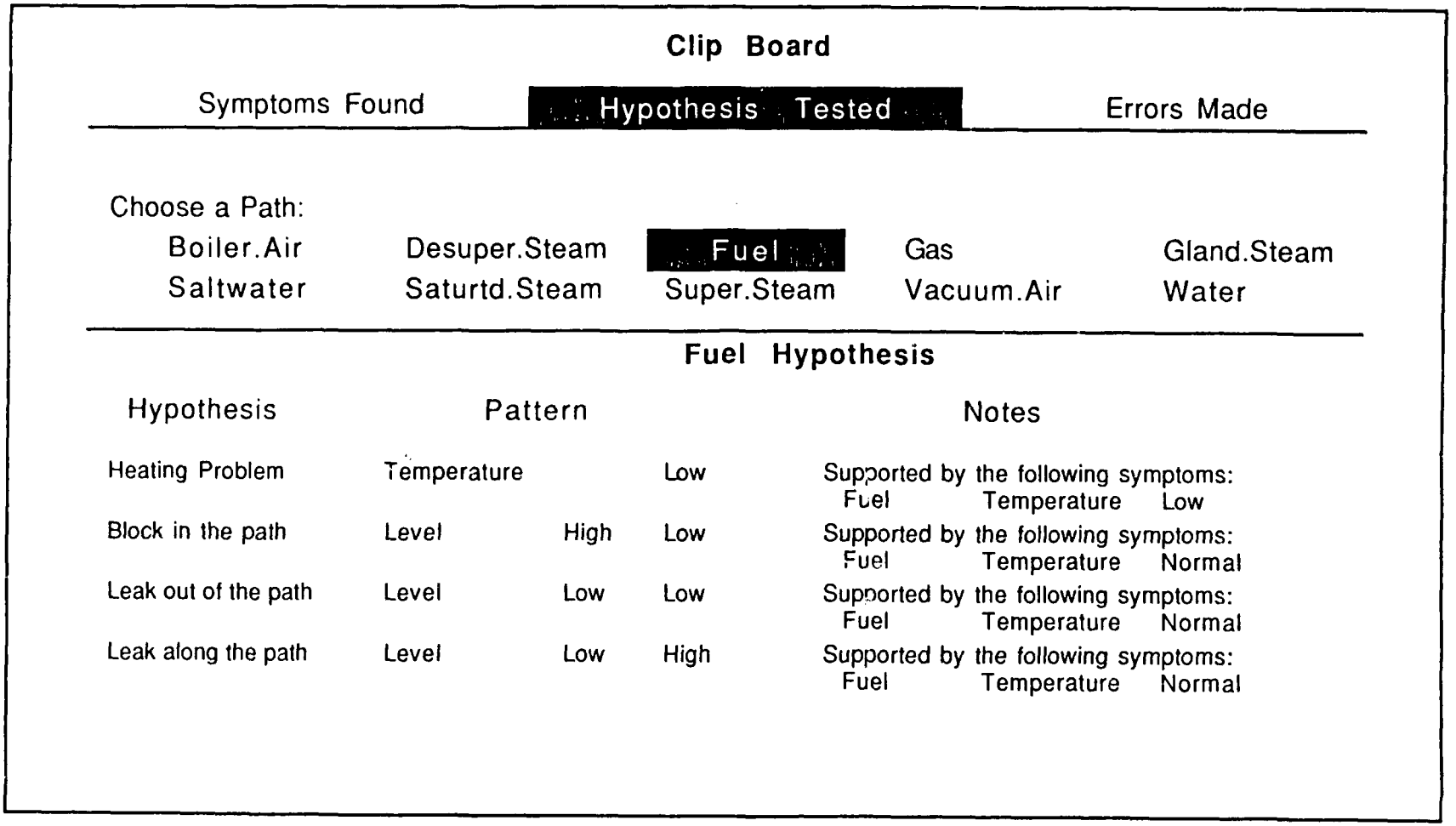


Figure 12. Example Clipboard List of Hypotheses Tested

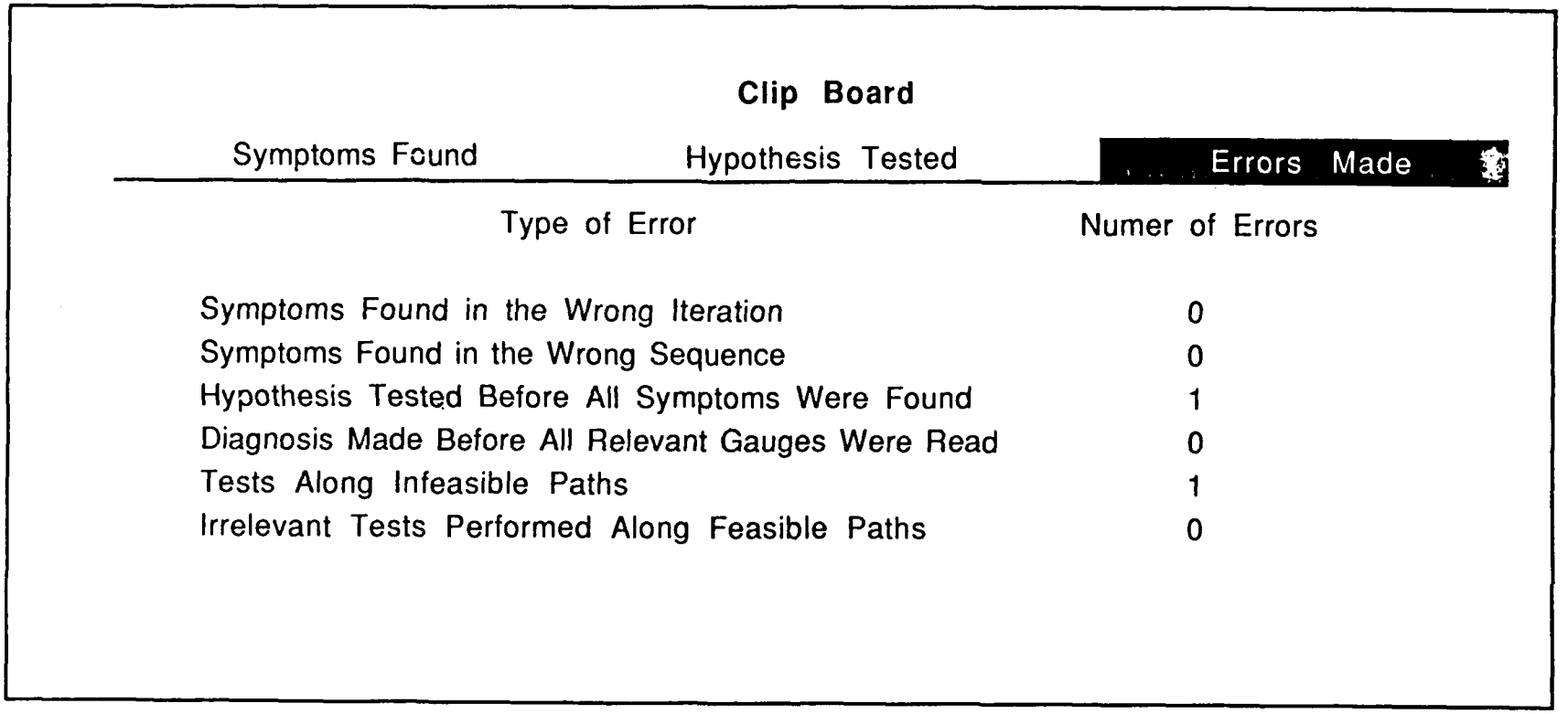


Figure 13. Example Clipboard List of Errors

Clip Board

Symptoms Found | Hypothesis Tested | Errors Made

Type of Error	Numer of Errors
Symptoms Found in the Wrong Iteration	0
Symptoms Found in the Wrong Sequence	0
Hypothesis Tested Before All Symptoms Were Found	1
Diagnosis Made Before All Relevant Gauges Were Read	0
Tests Along Infeasible Paths	1
Irrelevant Tests Performed Along Feasible Paths	0

You observed the Water Level even though the Water path was no longer feasible.

Figure 14. Example of General and Specific Errors Made from the Clip Board

Figure 15. Results for Average Number of Actions

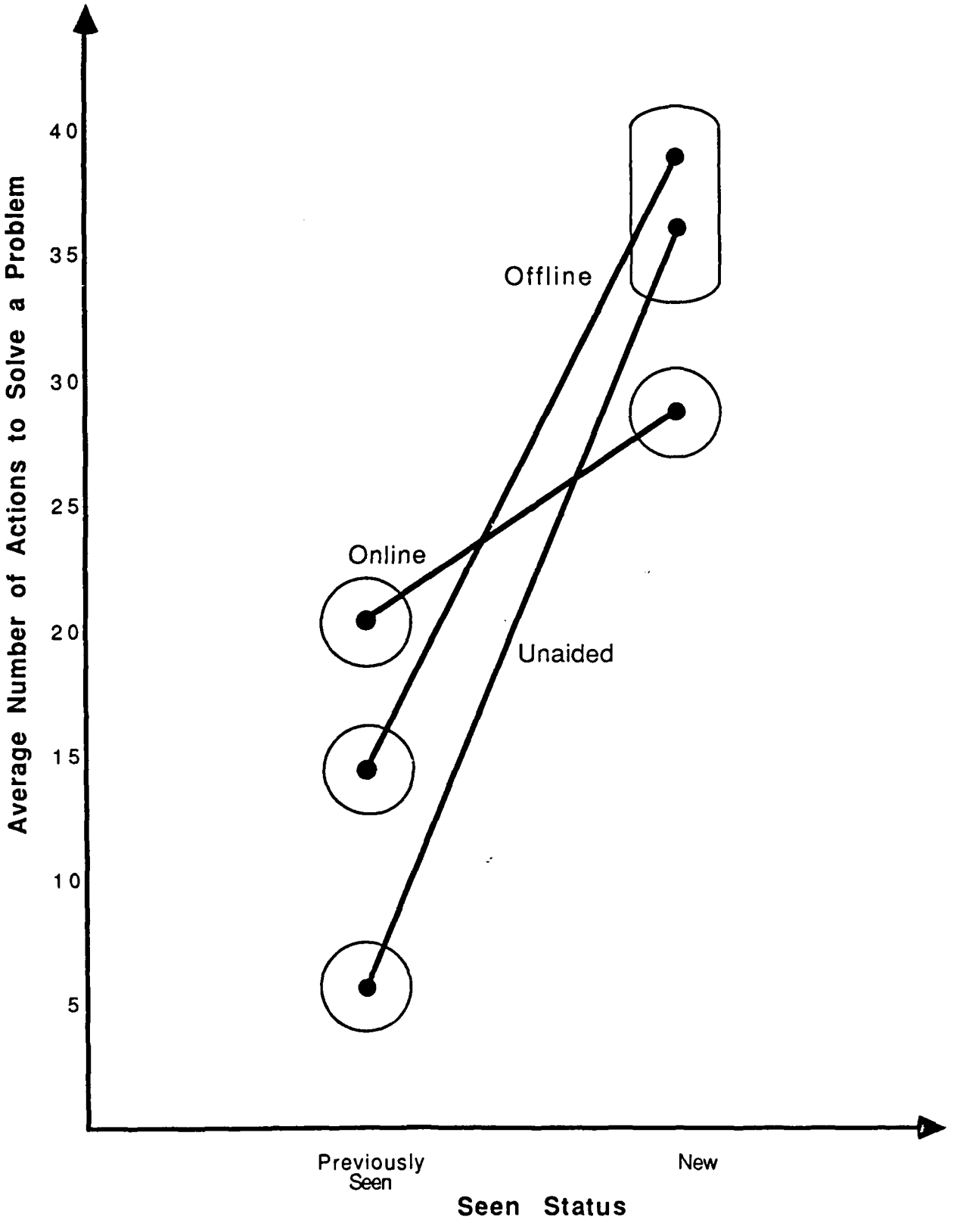
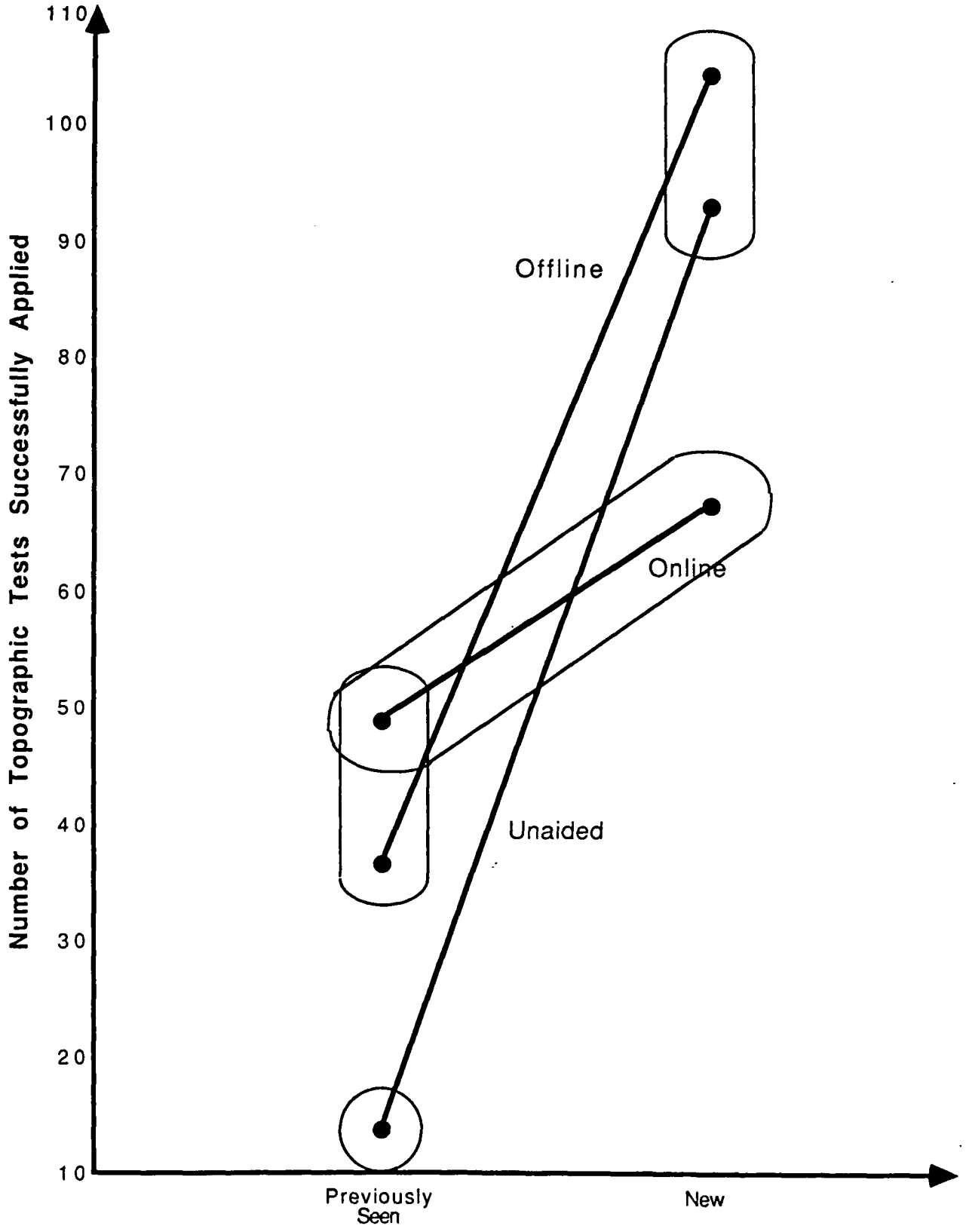


Figure 16. Results for Average Number of Topographic Tests



Manpower, Personnel, and Training R&D Program

One copy to each addressee except as otherwise noted:

Director Research Programs
Office of Naval Research (Code 11)
Arlington, VA 22217-5000

Chairman, MPT R&D Committee
Office of the Chief of Naval Research
Code 222
Arlington, VA 22217-5000

Mathematics (Code 1111MA)
Office of Naval Research
Arlington, VA 22217-5000

Director, Life Sciences (Code 114)
Office of Naval Research
Arlington, VA 22217-5000

Director, Cognitive & Neural Sciences
(Code 1142)
Office of Naval Research
Arlington, VA 22217-5000

Cognitive Science (Code 1142CS)
Office of Naval Research
Arlington, VA 22217-5000

Perceptual Science (Code 1142PS)
Office of Naval Research
Arlington, VA 22217-5000

Biological Intelligence (Code 1142BI)
Office of Naval Research
Arlington, VA 22217-5000

Director, Applied Research
Division (Code 121)
Office of the Chief of Naval Research
Arlington, VA 22217-5000

Defense Technical Information Center
(12 copies)*
DTIC/DDA-2
Cameron Station, Building 5
Alexandria, VA 22314

Science and Technology Division
Library of Congress
Washington, DC 20540

Commanding Officer
Naval Research Laboratory
Code 2627
Washington, DC 20375

Psychologist
Office of Naval Research Detachment
1030 East Green Street
Pasadena, CA 91106-2485

Office of the Deputy Assistant Secretary
of the Navy (Manpower & Reserve Affairs)
5D800, The Pentagon
Washington, DC 20350-1000

Assistant for Long Range Requirements
CNO Executive Panel (Op-00K)
4401 Ford Avenue
Alexandria, VA 22302-0268

Head, Manpower, Personnel, and
Training Branch
Office of the CNO (Op-813)
4A478, The Pentagon
Washington, DC 20350-1000

Assistant for Manpower and Training
Office of the CNO (Op-987H)
5D772, The Pentagon
Washington, DC 20350

Assistant for Planning and Technology
Development
Office of the DCNO(MPT) (Op-01B2)
Department of the Navy
Washington, DC 20350-2000

Deputy Director Total Force Training
and Education Division
Office of the DCNO(MPT) (Op-11B)
Department of the Navy
Washington, DC 20370-2000

---

* If report is ready for unlimited public distribution

Deputy Director Military Personnel
Policy Division
Office of the DCNO(MPT) (Op-13B)
Department of the Navy
Washington, DC 20370-2000

Head, Military Compensation Policy Branch
Office of the DCNO(MPT) (Op-134)
Department of the Navy
Washington, DC 20370-2000

Director, Navy Family Support Program
Office of the DCNO(MPT) (Op-156)
1300 Wilson Boulevard, Room 828
Arlington, VA 22209

Headquarters U.S. Marine Corps
Code MA
Washington, DC 20380-0001

Head, Leadership Branch
Naval Military Personnel Command
Attn: LCDR E. Marits, NMPC-621
Department of the Navy
Washington, DC 20370-5620

Director, Recreational Services Department
Naval Military Personnel Command (N-651C)
1300 Wilson Boulevard, Room 932
Arlington, VA 22209

Deputy Director Manpower, Personnel
and Training Division
Naval Sea Systems Command
Attn: Code CEL-MP63
Washington, DC 20362

Director, Research & Analysis Division
Navy Recruiting Command (Code 223)
4015 Wilson Boulevard, Room 215
Arlington, VA 22203-1991

Naval School of Health Sciences
National Naval Medical Center (Bldg. 141)
Washington, DC 20814-5033
Attn: CDR J. M. LaRocco

Technical Director
Naval Health Research Center
P.O. Box 85122
San Diego, CA 92138-9174

Deputy Director, R&D Department
Naval Training Systems Center (Code 7A)
Orlando, FL 32813-7100
Attn: David E. Daniel

Head, Human Factors Laboratory
Naval Training Systems Center (Code 71)
Orlando, FL 32813-7100

Human Factors Division (Code 712)
Naval Training Systems Center
Orlando, FL 32813-7100
Attn: Dr. Eduardo Salas

Commanding Officer
Navy Personnel R&D Center
San Diego, CA 92152-6800

Technical Director
NPRDC (Code 01)
San Diego, CA 92152-6800

Head, Fleet Liaison Department
NPRDC (Code 31)
San Diego, CA 92152-6800

Head, Human Factors Department
NPRDC (Code 41)
San Diego, CA 92152-6800

Head, Training Technology Department
NPRDC (Code 51)
San Diego, CA 92152-6800

Head, Training Systems Department
NPRDC (Code 52)
San Diego, CA 92152-6800

Head, Manpower Systems Department
NPRDC (Code 61)
San Diego, CA 92152-6800

Head, Personnel Systems Department
NPRDC (Code 62)
San Diego, CA 92152-6800

Head, Testing Systems Department
NPRDC (Code 63)
San Diego, CA 92152-6800

Chairman, Department of Administrative Sciences (Code 54)
Naval Postgraduate School
Monterey, CA 93943-5100

Chairman, Department of Operations Research (Code 55)
Naval Postgraduate School
Monterey, CA 93943-5100

Director, Instructional Development and Educational Program Support Department
Naval Education and Training Program Management Support Activity (NETPMSA)
Pensacola, FL 32509

Academic Programs and Research Branch
Naval Technical Training Command
Code N-625
NAS Memphis (75)
Millington, TN 38054

Assistant for Training and Personnel Technology
Office of the Under Secretary of Defense for Research and Engineering
3D129, The Pentagon
Washington, DC 20301-3080

Director, Defense Personnel Security Research and Education Center
Suite E, Building 455
99 Pacific Street
Monterey, CA 93940-2481

Personnel Analysis Division
AF/DPXA
5C360, The Pentagon
Washington, DC 20330

Technical Director
U.S. Army Research Institute for the Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Director, Manpower Program
Center for Naval Analyses
4401 Ford Avenue
P.O. Box 16268
Alexandria, VA 22302-0268

Technical Director
Air Force Human Resources Laboratory
Brooks Air Force Base, TX 78236-5601

Library
Naval Training Systems Center
Orlando, FL 32813

Library
Naval War College
Newport, RI 02940

Chief, Survey and Market Analysis Division
Defense Manpower Data Center
1600 Wilson Boulevard, #400
Arlington, VA 22209

Program Director
Manpower Research & Advisory Services
Smithsonian Institution
801 North Pitt Street, Suite 120
Alexandria, VA 22314-1713

Dr. Meg Gerrard
Psychology Department
Iowa State University
Ames, Iowa 50011

Dr. Perry W. Thorndyke
FMC Central Engineering Labs
Box 580
Santa Clara, CA 95052

Dr. T. Govindaraj
School of Industrial & Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205

Prof. David W. Johnson
Cooperative Learning Center
University of Minnesota
150 Pillsbury Drive, S.E.
Minneapolis, MN 55455

Dr. Walter Schneider
Learning Research & Development Center
University of Pittsburgh
Pittsburgh, PA 15620

Prof. George A. Miller
Department of Psychology
Princeton University
Princeton, NJ 08544

Dr. Jeffery L. Kennington
School of Engineering & Applied Science
Southern Methodist University
Dallas, TX 75275-0335

Prof. Clark Glymour
Department of Philosophy
Carnegie-Mellon University
Pittsburgh, PA 15213

Prof. Kent E. Williams
Institute for Simulation & Training
University of Central Florida
P.O. Box 25000
Orlando, FL 32816-0544

Dr. Paul Feltovich
Southern Illinois University
School of Medicine
P.O. Box 3926
Springfield, IL 62708

Prof. Thomas G. Bever
Department of Psychology
The University of Rochester
River Station
Rochester, NY 14627

Dr. Lawrence J. Stricker
Educational Testing Service
Princeton, NJ 08541

Prof. Michael Levine
Dept. of Educational Psychology
University of Illinois
506 South Wright St.
Urbana, IL 61801

Prof. Patricia A. Carpenter
Psychology Department
Carnegie-Mellon University
Pittsburgh, PA 15213

Dr. William B. Johnson
Search Technology, Inc.
4725 Peachtree Corners Circle
Norcross, GA 30092